ملخصات شوم
نظريات ومسائل
في

البرمجة بلغة البيسك

يتضمن قسم موسع لبيسك الحاسب الدقيق

تأليف

بايرون س. جوتفريد ، Ph. D.
أستاذ في الهندسة الصناعية
هندسة إدارة النظم وبحوث العمليات
جامعة بيتسبرج

ترجمة

ابتسام صديق أبو الخير

ماجستير حسابات علمية
نائب مدير مركز بحوث الحسابات العلمية والاحصائية
جامعة القاهرة ــ جمهورية مصر العربية

مراجعة

الأستاذ الدكتور/ أحمد عزيز كمال

عميد كلية الهندسة ــ جامعة القاهرة
ومدير مركز بحوث الحسابات العلمية والاحصائية
جامعة القاهرة

قام بترجمة الاضافات لهذه الطبعة الجديدة

دكتور / عباده سرحان
أستاذ مساعد بكلية الهندسة والتكنولوجيا
ومدير مركز الحساب العلمي
جامعة حلوان

الدار الدولية للنشر والتوزيع

هــذا الكتــاب

صدرت الطبعة الثالثة من هذا الكتاب باللغة الإنجليزية وتشمل إضافات جديدة كان لابد من ترجمتها خدمة للقارئ العربى .

وقد تولت الدار الدولية للنشر والتوزيع مهمة طبع ونشر وتوزيع هذه الترجمة الشاملة الجديدة انطلاقًا من رسالتها فى نشر كل جديد .

الدار الدولية للنشر والتوزيع
سبتمبر ١٩٨٩

٥١٣١٩

البرمجة بلغة البيسك

حقوق النشر

● الطبعة الإنجليزية : حقوق التأليف © ١٩٨٦ ، ١٩٨٢ ، ١٩٧٥ دار ماكجروهيل للنشر ، إنك ، جميع الحقوق محفوظه .

Schaum's Outline of Theory and Problems of
PROGRAMMING WITH BASIC 3/ed
Byron S. Gottfried
ISBN 0-07-023875-8

● الطبعة العربية الأولى : حقوق الطبع والنشر © ١٩٨٤ دار ماكجروهيل للنشر ، إنك — جميع الحقوق محفوظه .

● الطبعة العربية الثانية : حقوق الطبع والنشر © ١٩٨٨ ، جميع الحقوق محفوظة للناشر :

● الطبعة العربية الثالثة : حقوق الطبع والنشر © ١٩٨٩ ، جميع الحقوق محفوظة للناشر :

الدار الدولية للنشر والتوزيع

ص . ب ٥٥٩٩ هليوبوليس غرب — القاهرة

ت : ٢٥٨٢٨٨٧

تلكس : ٢٠٠٧٠ PBCRB UN

فاكس : ٢٩١٨٠٥٩ /٠٠٢٠٢

لا يجوز نشر أى جزء من هذا الكتاب أو اختزان مادته بطريقه الاسترجاع أو نقله على أى وجه أو بأى طريقه سواء كانت اليكترونيه أو ميكانيكية أو بالتصوير أو بالتسجيل أو خلاف ذلك إلا بموافقة الناشر على هذا كتابة ومقدماً .

مقدمة الناشـــر

المعرفة هى أصل الحضارة ،
والكلمة هى أصل المعرفة ،
والكلمة المطبوعة هى أهم مكون فى هذا المصدر .

وقد كانت الكلمة المطبوعة ولا تزال أهم وسائل الثقافة والإعلام وأوسعها انتشارًا وأبقاها أثرًا ، حيث حملت إلينا حضارات الأمم عبر السنين لتتولى الأجيال المتلاحقة صياغة حضارتها وإضاءة الطريق بنور العلم والمعرفة .

والكلمة تبقى مجرد فكرة لدى صاحبها حتى تتاح لها فرصة نشرها وترجمتها إلى لغات الآخرين ، ثم توزيعها ، وذلك وحده هو الذى يكفل لها أداء رسالتها .

وعالم الكتب العلمية عالم رحب ممتد الآفاق ، متسع الجنبات ، والعلم لاوطن له ولا حدود ، ويوم يحظى القارئ العربى بأحدث الكتب العلمية باللغة العربية لهو اليوم الذى تتطلع له الأمة العربية جمعاء .

والدار الدولية للنشر والتوزيع تشعر بالرضا عن مساهمتها فى هذا المجال بتقديم الطبعات العربية للكتب العلمية مستهدفة توفير احتياجات القارئ العربى أستاذاً وباحثاً وممارساً .

والله ولى التوفيق ،،،

محمد وفائى كامل

المقدمة

لغة البيسك تعتبر من أسهل اللغات الشائعة حالياً فى التعليم والاستخدام . وتحتوى هذه اللغة مع بساطتها الملحوظة بقوة كافية واستعمالات متعددة يمكن أن يستفيد منها العديد من الناس المختلفين ولأنواع عديدة من التطبيقات . وتبعاً لذلك فإن دروس لغة البيسك BASIC أصبحت شائعة فى عديد من المدارس الثانوية وكذلك المدارس الإعدادية بالإضافة إلى معظم الكليات والجامعات . والآن المدارس الأولية بالولايات المتحدة الأمريكية تعرض مقدمة لدروس البيسك لمجموعات منتقاة من الطلبة .

سبب آخر لشعبية البيسك هو انتشاره الواسع ؛ فاللغة الآن متاحة عموماً على كل الحاسبات الكبيرة ، ومدعمة فعليا بواسطة كل الخدمات للمشاركة الزمنية التجارية . وعلاوة على ذلك ، فإن البيسك أصبح اللغة الأساسية لمعظم تطبيقات الحاسبات الدقيقة . ولذلك فإن اللغة يمكن استخدامها على كل الحاسبات الكبيرة والصغيرة وفى كل أنواع بيئات البرمجة المختلفة .

ويعرض هذا الكتاب تعليمات فى برمجة الحاسب باستخدام الخصائص القياسية للغة البيسك وقد تمت مناقشة كل الخصائص الأساسية للغة البيسك . وبالإضافة إلى ذلك ، فإن الكتاب يبرز التطور فى البرامج من حيث المنطق والكفاءة والنظام . ولذلك فإن القارىء يتعرض لأساسيات التدريب على البرمجة الجيدة بجانب القواعد المحددة فى لغة البيسك .

طريقة الكتابة بطريقة أولية وبصورة متعمدة . وجعلت الكتاب سهل الفهم بواسطة قاعدة عريضة من القراء المستمعين يتراوح بين طلبة المدارس الثانوية والموظفين . ويناسب هذا الكتاب مستوى المدارس الثانوية المتقدمة أو المستوى الجامعى للطلبة المبتدئين ككتاب أساسى فى محاضرات البرمجة وكنص مكمل لمحاضرات أكثر شمولا فى الطرق الفنية التحليلية أو كمرشد فعال للدراسة بدون معلم . وفى معظم أجزائه مستوى الرياضيات المطلوبة لاتتجاوز جبر المدارس العليا .

وفى هذه الطبعة الثالثة تم التوسع كثيراً فى المادة الخاصة ببيسك الحاسب الدقيق ، وتم تنظيم الكتاب فى ثلاثة أجزاء رئيسية هى : البيسك الأولى ، البيسك المتقدم ، وبيسك الحاسب الدقيق . ويمثل الجزء الأول — خصائص اللغة شائعة الاستخدام . ويمكن تدريس مقرر مختصر فى البرمجة فى هذه المادة وحدها . ويتعلق الجزء الثانى بخصائص أكثر تخصصا مثل البرامج وجمل المصفوفات ، والتعامل مع الملف . ويركز الجزء الثالث على التحسينات المتاحة فى سبيل الحاسب الدقيق ، مع تأكيد خاص على ميكروسوفت بيسك ، كما ينفذ على حاسب IBM الشخصى . وتحتوى هذه المادة على فصل مختصر فى البرمجة سهلة الاستخدام ، وفصل أطول على بيانات الحاسب الدقيق .

تقدم المادة بطريقة تمكن القارىء من كتابة برامج بيسك كاملة ولكن بسيطة بأسرع ما يمكن . ومن المهم أن يكتب القارىء مثل هذه البرامج ويشغلها على الحاسب فى نفس الوقت الذى يدرس فيه الكتاب. وهذا يزيد من ثقة المبرمج الحديث فى نفسه وتثير من تحمسه لهذا الموضوع (تعلم كيفية برمجة الحاسب كتعلم كيفية العزف على الة موسيقية . لايمكن تعلمها بسهولة بدراسة كتاب !) .

يتضمن هذا الكتاب عدداً كبيراً من الأمثلة كجزء مكمل له . وهذه تتضمن عدداً من مسائل البرمجة الشاملة بجانب أنواع التدريبات العادية . بالإضافة إلى ذلك ، تتضمن نهاية معظم الفصول مجموعة من المسائل المحلولة . ويجب دراسة هذه الأمثلة والمسائل المحلولة بكل عناية أثناء تقدم القارىء فى دراسته لفصول الكتاب وتدربه على كتابة البرامج الخاصة به .

وتتضمن أيضاً مجموعات من أسئلة المراجعة والمسائل التكميلية ومسائل للبرمجة فى نهاية كل فصل . تمكن أسئلة المراجعة القارىء من اختبار استرجاعه للمواد المقدمة فى الفصل . وتمدنا أيضاً بملخص مؤثر للفصل . ولا تتطلب معظم المسائل التكميلية ومسائل البرمجة أى خلفية خاصة فى الرياضيات أو التكنولوجيا . ويجب على الطالب أن يحل هذه المسائل بقدر المستطاع (الإجابة على هذه المسائل التكميلية أيضاً متاحة فى نهاية الكتاب) . وعند استخدام هذا الكتاب كنص محاضرات ، فإننا ننصح القائم بالتدريس بأن يكمل مسائل البرمجة ببرامج إضافية تتلاءم مع اهتمامات الطالب الخاصة .

الخصائص الأساسية المهمة للغة ملخصة داخل الغلاف الخلفى وفى خمسة ملاحق وذلك لتسهيل مهمة القارىء . يجب أن تستخدم هذه المادة باستمرار كمرجع جاهز واسترجاع سريع . وسوف تكون مفيدة عند كتابة أو متابعة البرنامج الجديد .

وأخيراً ، فإن القارىء الذى يكمل هذا الكتاب سيكون قد تعلم الكثير عن مفاهيم لغات الحاسب العامة بجانب القواعد الخاصة بلغة لبيسك ويجب أن يكون مقتنعاً تماماً أن البرمجة بلغة البيسك ليست سهلة فقط ولكنها مسلية .

بايرون س . جوتفريد

المحتويـــات

صفحـــة

صفحة

صفحـــة

أمثلة لمجموعة برامج كاملة :

(١) جذور معادلة تربيعية (أمثلة ٢ ــ ٢٦ ، ٢ ــ ٣٠) ــ ٤٥ ، ٤٧
(٢) مساحة ومحيط دائرة (مثال ٣ ــ ٨ ، ٣ ــ ٩) ــ ٧٠
(٣) جذور معادلة جبرية (مثال ٤ ــ ٥) ــ ٨٣
(٤) حساب قيمة الاستهلاك (مثال ٤ ــ ٩) ــ ٨٨
(٥) إيجاد متوسط بيانات تلوث الهواء (مثال ٤ ــ ١٦) ــ ٩٥
(٦) توليد أرقام فيبوناسى والبحث عن الأرقام الأولية (مثال ٤ - ١٨) - ١٠٠
(٧) جدول للدوال (مثال ٥ ــ ٥) ــ ١١٧
(٨) كلمة غير مرتبة (مثال ٥ - ٩) ــ ١٢٠ .
(٩) إعادة ترتيب قائمة من الأرقام (مثال ٥ ــ ١٤) ــ ١٢٦
(١٠) معالجة عناصر الجدول (مثال ٥ - ١٥) ــ ١٢٩
(١١) البحث عن اكبر قيمة (مثال ٦ - ٦) ــ ١٥٣
(١٢) مولد بجلاتين (مثال ٦ - ١٥) ــ ١٦٣
(١٣) لعبة حظ (اصطياد كرابس) (مثال ٦ - ٢٠) ــ ١٦٨
(١٤) برنامج المرتبات الشهرية (مثال ٦ - ٢٦) ــ ١٧٦
(١٥) محاكاة إرتداد كره (مثال ٦ - ٢٨) ــ ١٨٤
(١٦) التعامل مع مصفوفة (مثال ٧ - ١٥) ــ ٢١٤
(١٧) معكوس المصفوفة (مثال ٧ - ١٨) ــ ٢٢٠
(١٨) المعادلات الآنية (مثال ٧ - ١٩) ــ ٢٢٢
(١٩) توفيق منحنى المربعات الصغرى (لعبة الاسواق المالية) (مثال ٧ - ٢٢) ــ ٢٢٥
(٢٠) تشغيل درجات اختبار طالب (مثال ٨ - ٤) ــ ٢٤٧
(٢١) مراتبة المخزون (مثال ٨ - ١٠) ــ ٢٥٨
(٢٢) البحث فى ملف بيانات (مثال ٨ - ١٣) ــ ٢٦١
(٢٣) توليد أرقام فيبوناسى والبحث عن الارقام الأولية وتشغيلها على حاسب دقيق (مثال ٩ - ١١) ــ ٢٧٩
(٢٤) البحث عن القيمة العظمى على حاسب دقيق (مثال ٩ - ٢٧) ــ ٢٨٨
(٢٥) توليد بجلاتين على حاسب دقيق (مثال ٩ - ٢٨) ــ ٢٩٢
(٢٦) خلق ملف بيانات متتالى (مثال ٩ - ٢٩) ــ ٢٩٧
(٢٧) تشغيل درجات إختبار طالب على حاسب دقيق (مثال ٩ - ٣٠ ، ١٠ ــ ٢٠) ــ ٣٠٠ ، ٣٣١
(٢٨) التحكم فى المخزون على حاسب دقيق (مثال ٩ - ٣١) ــ ٣٠٠
(٢٩) زمن اليوم (مثال ١٠ - ٥) - ٣١٢ .
(٣٠) برمجة عرض على شاشة تليفزيونية (لا شيء يمكن أن يصبح خطأ .. خطأ .. خطأ ..) (مثال ١٠ - ١٠ ، ١٠ - ١٧) - ٣١٦ ، ٣٢٨
(٣١) برمجة مفاتيح الدوال (مثال ١٠ - ١١) ــ ٣١٨
(٣٢) برمجة قلم ضوئى (مثال ١٠ - ١٢) ــ ٣٢٠
(٣٣) برمجة عصا توجيه (مثال ١٠ - ١٣) ــ ٣٢٣
(٣٤) النص متعدد الألوان (مثال ١٠ - ١٤) ــ ٣٢٥
(٣٥) برمجة البوق (صفارة انذار) (مثال ١٠ - ١٦) ــ ٣٢٧
(٣٦) ادخال درجات اختبار طالب (مثال ١١ - ٢ ، ١١ - ٦) ــ ٣٣٨ ، ٣٥٠
(٣٧) التمويل الشخصى (حسابات الربح المركب) (مثال ١١ - ٤) - ٣٤١ .
(٣٨) تخزين بيانات المعامل (مثال ١١ ــ ٧) ــ ٣٥٣
(٣٩) نقط فى الفراغ (مثال ١٢ - ٨) - ٣٦٣ .

الجزء الأول : أساسيات لغة البيسك

الفصـل ١

مفاهيم تمهيدية .

Introductory Concepts

يعرض هذا الكتاب تعليمات لبرمجة الحاسب باستخدام لغة برمجة سهلة فى تعلمها وشائعة تسمى بيسك (Beginner's All-purpose Symbolic (BASIC) Instruction Code) وسوف نرى كيف يمكن تحليل مسألة وصفت مبدئياً بالكلمات وتحولت أخيراً إلى برنامج بيسك قابل للتشغيل . وقد عرضت هذه المفاهيم بصورة توضيحية ، بواسطة العديد من المسائل المختارة .

١ - ١ مقدمة للحاسبات INTRODUCTION TO COMPUTERS

توجد الحاسبات حالياً فى أشكال وأحجام وأسعار مختلفة وتقوم العديد من المؤسسات الكبيرة والجامعات والمستشفيات والمصالح الحكومية باستخدام الحاسبات الضخمة ذات الأغراض العامة للقيام بالحسابات التجارية والعلمية المعقدة . وتعرف هذه بالحاسبات الكبيرة وهى غالية الثمن جداً (بعضها يبلغ ثمنه ملايين الدولارات) وتحتاج إلى بيئة يمكن التحكم فيها بعناية (درجة الحرارة والرطوبة ... الخ) وكقاعدة عامة فإنه من غير المسموح لمن يستخدموها التعامل معها مباشرة .

وهذه الحاسبات الكبيرة أصبحت متاحة منذ أوائل الخمسينات رغم إنه لم يكن هناك إلا عدد قليل من الأفراد يعلمون كيفية استعمالها فى السنوات الأولى . هؤلاء الأفراد والمحظوظين الذين استخدموها كانوا عموماً من العلماء والمهندسين والمحاسبين وعلى درجة عالية من التدريب ، وعلى ذلك فإنه لم يكن غريباً أن ينظر عامة الناس للحاسبات بنظرة الشك والرهبة . وفى خلال الستينيات أصبح من المعتاد أن يقوم الطلاب فى الجامعات بتعلم كيفية برمجة الحاسبات الكبيرة (عادة باستخدام لغة Basic أو أى لغة برمجة أخرى من المستوى الرفيع) ونتيجة لذلك فقد بدأ يختفى بعض الغموض المرتبط باستخدام الحاسبات .

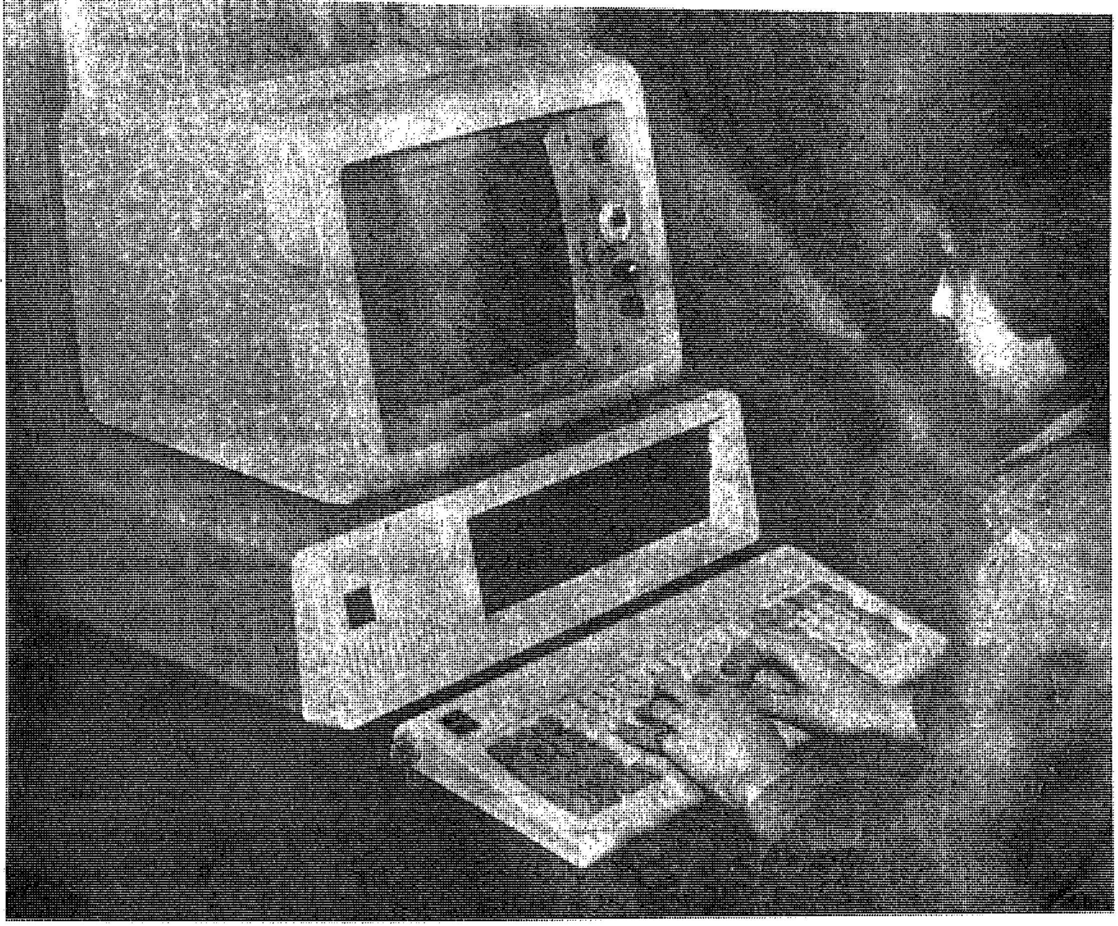
وقد شاهدت أواخر الستينات وأوائل السبعينات ظهور حاسبات صغيرة أقل تكلفة وأصغر حجماً وهذه الأجهزة تعطى كفاءة الحاسبات الكبيرة السابقة بجزء من تكلفتها فقط ، مما أمكن معه للعديد من المدارس والمؤسسات التجارية الحصول على حاسبات صغيرة عندما أصبحت متاحة ، والتى لم يكن فى امكانها الحصول على حاسبات كبيرة . وقد نتج عن التقدم فى تكنولوجيا الدوائر المتكاملة (شرائح السليكون) الوصول إلى حاسبات أصغر كثيراً وأقل تكلفة تسمى الحاسبات الدقيقة . ولما كانت هذه الأجهزة قد بنيت بالكامل من الدوائر المتكاملة ولذا لم تكن أكبر كثيراً أو أغلى كثيراً من آلة كاتبة عادية .

وعادة ما يشار إلى الحاسبات الدقيقة على إنها حاسبات شخصية حيث أن معظمها يستخدم بواسطة شخص واحد فى لحظة واحدة ، وشكل (١-١) يبين طالب يستخدم حاسب شخصى .

وهناك العديد من الحاسبات الدقيقة الحديثة تقترب قوتها الحسابية من الحاسبات الصغيرة ، وتستمر فى تحسين كفاءتها بشكل كبير فى الوقت الذى تقل أسعارها باستمرار ، وعلى ذلك فإنه يمكن استخدامها فى العديد من التطبيقات الفنية والتجارية والتعليمية والشخصية ولذلك فإننا نجد أن الحاسبات الدقيقة منتشرة بكثرة فى العديد من المدارس والمؤسسات التجارية ، وعلى ما يبدو فإنها ستنتشر قريباً كأحدالوحدات المنزلية الشائعة ، وهناك العديد من المؤسسات الكبيرة التى تستخدم الحاسبات الدقيقة كنهائيات طرفية أو محطات عمل ويتم توصيلها بحاسبات أكبر أو حاسبات دقيقة أخرى عن طريق شبكة اتصالات ، وعندما تستخدم بهذه الطريقة فإن الحاسبات الدقيقة تميل إلى اتمام الاستخدامات على الحاسبات الكبيرة بدلاً من أن تحل محلها .

١ - ٢ خصائص الحاسب COMPUTER CHARACTERISTICS

ومهما تكن أحجام الحاسبات الرقية ، فهى تتكون أساساً من وحدات إلكترونية يمكنها إرسال وتخزين وتداول المعلومات (أى البيانات) . ويوجد نوعان مختلفان من البيانات وهما : بيانات **رقية** وبيانات **هجائية** (مثل الأسماء والعناوين ..إلخ) .



شكل ١ – ١

تتطلب التطبيقات العلمية والفنية أساساً تشغيل بيانات رقية ، بينما تتضمن التطبيقات التجارية عادة تشغيل كل من البيانات الهجائية والرقية . وتستخدم بعض الحاسبات فقط لتشغيل البيانات الهجائية التى ترد فى النصوص المكتوبة (مثل الخطابات ومخطوطات الكتب ..إلخ) ، وتعرف هذه بمعالجة الكلمات .

ومن أجل تشغيل مجموعة معينة من البيانات فيجب أن يعطى الحاسب مجموعة صحيحة من التعليمات تسمى برنامج . ويتم ادخال هذه التعليمات فى الحاسب ثم يتم تخزينها فى جزء من ذاكرة الحاسب .

وفى أى وقت يتم فيه تنفيذ برنامج سبق تخزينه يحدث ما يلى :

١ – يتم إدخال مجموعة من المعلومات ، تسمى **المدخلات** (من خلال نهاية طرفية مثل الآلة الكاتبة أو من خلال قارئ البطاقات أو .. إلخ) ويتم تخزينها فى جزء آخر من ذاكرة الحاسب .

٢ – بعد ذلك يتم تشغيل هذه المدخلات لإصدار نتائج معينة مطلوبة تسمى **بالمخرجات** .

٣ – ثم تطبع المخرجات (وربما جزء من المدخلات) على فرخ من الورق أو تعرض على شاشة مرئية TV .

ويمكن تكرار هذا الإجراء ذى الخطوات الثلاث عدة مرات إذا تطلب الأمر ذلك ، وبذا يتم تشغيل كية كبيرة من البيانات فى توال سريع . ويجب أن يكون مفهوماً أن كل هذه الخطوات وبالذات الخطوة رقم ٢ ورقم ٣ يمكن أن تكون طويلة ومعقدة .

مثال ١ – ١

تم كتابة برنامج لحاسب لحساب مساحة دائرة باستخدام الصيغة الرياضية $A = \pi r^2$ عند إعطاء قيمة رقية لنصف القطر r كدخل ، وسوف يتضمن التنفيذ الخطوات التالية :

١ – قراءة القيمة الرقية لنصف قطر الدائرة .

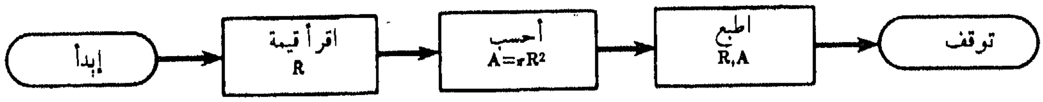
٢ – حساب قيمة المساحة باستخدام الصيغة الرياضية السابقة (وهذه القيمة سيتم تخزينها مع المدخلات فى ذاكرة الحاسب) .

٣ – طباعة (عرض) قيمة نصف القطر وقيمة المساحة المناظرة .

٤ – إيقاف التشغيل .

وكل خطوة من هذه الخطوات سنحتاج إلى تعليمة واحدة أو أكثر فى برنامج الحاسب .

يمكن تمثيل جميع الإجراءات بطريقة تصويرية كالمبينة فى شكل ١ – ٢ . وتعرف بخريطة سير العمليات . تساعد خرائط سير العمليات القارئ فى تصور طريقة انسياب المنطق فى البرنامج .



شكل ١ – ٢

توضح المناقشة السابقة خاصيتين مهمتين من خواص الحاسب الرقى وهما : **الذاكرة والمقدرة على البرمجة** . وخاصية ثالثة أخرى مهمة ألا وهى السرعة **والاعتمادية** . وسوف نذكر الكثير عن الذاكرة والسرعة والاعتمادية فى الفقرات القليلة القادمة . أما موضوع البرمجة فسوف يتم مناقشته بالتفصيل خلال الأجزاء الباقية من هذا الكتاب .

الذاكرة **Memory**

يكود كل جزء من المعلومات التى تخزن فى ذاكرة الحاسب كتوافقية فريدة من الأرقام صفر وواحد تسمى هذه الأرقام من صفر وواحد أرقاماً ثنائية *bits* (*bi*nary digits) . يمثل كل رقم ثنائى بواسطة وحدة إلكترونية وهى إما مطفأة (صفر) أو « مضاءة » (واحد) .

معظم الحاسبات الصغيرة لها ذاكرة منظمة على أساس مضاعفات ثمانية أرقام ثنائية تسمى *bytes* (**بايتات**) . وعادة يمثل كل حرف ببايت واحد (حرف هجائى أو رقم مفرد أو رمز لعلامات الوصل) ويمكن أن يحتل الأمر بايتاً أو اثنين أو ثلاثة ، ويمكن أن تحتل الكمية الرقمية أى مكان من 2 إلى 8 بايت (معتمداً فى ذلك على نوع الرقم ومدى دقته) .

ويمكن عادة التعبير عن حجم ذاكرة الحاسب كبعض مضاعفات 2^{10} أى 1024 بايت . ويشار إليها (IK) . وتتراوح أحجام ذاكرة الحاسبات الصغيرة من 64 K إلى 1024 K بايت (IMB)

مثال ١ – ٢

سعة ذاكرة حاسب شخصى 256 K بايت وعلى ذلك يمكن تخزين 1024 x 256 يساوى 262,144 حرف أو تعليمه فى ذاكرة الحاسب . إذا استخدمت الذاكرة كلها لتمثل بيانات حرفية فيمكن تخزين حوالى 3200 اسم وعنوان بداخل الحاسب فى وقت واحد (وذلك بفرض 80 حرفاً لكل اسم وعنوان) .

أما إذا استخدمت الذاكرة لتمثيل بيانات رقية وليست أسماء وعناوين فيمكن تخزين حوالى 65,000 كية فى وقت واحد (بفرض 4 بايت لكل رقم) .

وتنظم ذاكرة الحاسبات الكبيرة فى صورة كلمات *words* وليست بايتات .وتتكون كل كلمة من عدد كبير نسبياً من الأرقام الثنائية ، والرقم النموذجى يتراوح ما بين 32 و 36 وذلك يسمح لكية رقية أو مجموعة صغيرة من الحروف (كنموذج 4 أو 5) أن تمثل بداخل كلمة واحدة فى الذاكرة . ويعبر عادة عن ذاكرة الحاسبات الكبيرة كمضاعفات (1 K) أى ($1024 = 2^{10}$) كلمة . ويكون للحاسبات الكبيرة ذاكرة بها عدة ملايين من الكلمات .

مثال ١ – ٣

ذاكرة حاسب كبير متعددة الأغراض لها سعة 2048 K والتى تساوى 1024×2048 أى 2,097,152 كلمة . إذا استخدمت الذاكرة كلها لتمثيل بيانات رقية فيمكن تخزين أكثر من 2 مليون رقم بداخل الحاسب فى وقت واحد .

أما إذا استخدمت الذاكرة لتمثيل حروف بدلا من بيانات رقية فيمكن تخزين حوالى 8 ملايين حرف فى وقت واحد . وتكون هذه الذاكرة كافية لتخزين محتويات كتاب بالكامل .

وتستخدم معظم الحاسبات وحدات ذاكرة مساعدة (من أمثلة ذلك شرائط مغناطيسية وأقراص وحدات ذاكرة صلبة) بالإضافة إلى ذاكرتها الأساسية . وتتراوح هذه الوحدات من بضع مئات الآلاف من البايت (لحاسب صغير) إلى عدة ملايين من الكلمات (لحاسب كبير) . وبالإضافة إلى ذلك فإن هذه الوحدات تسمح بتسجيل المعلومات تسجيلا دائماً ، حيث يمكن لها أن توضع على الحاسب أو تؤخذ وتخزن فى مكان آخر فى حالة عدم استخدامها . وبذلك فإن وقت التوصل (أى الوقت اللازم لتخزين أو استرجاع المعلومات) يكون أكثر بكثير لهذه الوحدات عنها للذاكرة الأساسية .

السرعة والاعتمادية Speed and Reliability

ولأن سرعة الحاسب فائقة فيمكنه أن يقوم بعمليات حسابية فى بضع دقائق يلزمها شهور أو ربما سنين إذا حسبت باليد . ويمكنه القيام بالمهام البسيطة مثل جمع رقين فى جزء صغير من ميكروثانية ($1\ \mu s = 10^{-6}$ s) . من الناحية العملية يمكن حساب درجات نهاية الفصل الدراسى لكل الطلاب الموجودين فى جامعة كبيرة فى بضع دقائق من وقت الحاسب .

ويصاحب هذه السرعة الفائقة مستوى عال مساو لها فى درجة الاعتمادية . ذلك أن الحاسب لا يرتكب عملياً أخطاء بنفسه . فأخطاء الحاسب المعلن عنها ، مثل تسلم شخص فاتورة شهرية بأكثر من مليون دولار من إحدى المحلات المحلية ، تكون كلها تقريباً نتيجة أخطاء برمجة أو خطأ فى إرسال البيانات .

١ – ٣ أساليب التشغيل 1.3 MODES OF OPERATION

يمكن الانتفاع بإمكانية الحاسب الرقى بطريقتين مختلفتين . هاتان الطريقتان هما أسلوب التشغيل على دفعات وأسلوب التشغيل التحاورى وكلاهما شائع . ولكل مزايا لأنواع معينة من المسائل .

التشغيل على دفعات Batch Processing

فى الأيام الأولى للعمليات الحسابية كان يتم التشغيل عن طريق التشغيل على دفعات ، ومازالت هذه الطريقة تستخدم فى بعض المدارس والمؤسسات التجارية إلا انها أقل شيوعاً عما كانت عليه سابقاً .

فى التشغيل على دفعات يتم قراءة عدد من الشغلات فى الحاسب وتخزن داخلياً ثم يتم تشغيلها على التوالى . (تشير الشغلة إلى برنامج الحاسب ومجموعة البيانات الملازمة له والمراد تشغيلها) وعادة يسجل البرنامج والبيانات على بطاقات مثقبة . تقرأ المعلومات المسجلة على بطاقات للحاسب بواسطة قارىء بطاقات ميكانيكى ، ثم يتم تشغيلها وبعد انتهاء تشغيل الشغلة ، تطبع النتائج مع قائمة للبرنامج على أفرخ كبيرة من الورق بواسطة آلة طباعة ذات سرعة عالية ، وهذا النوع من التشغيل على دفعات أصبح طراز قديم .

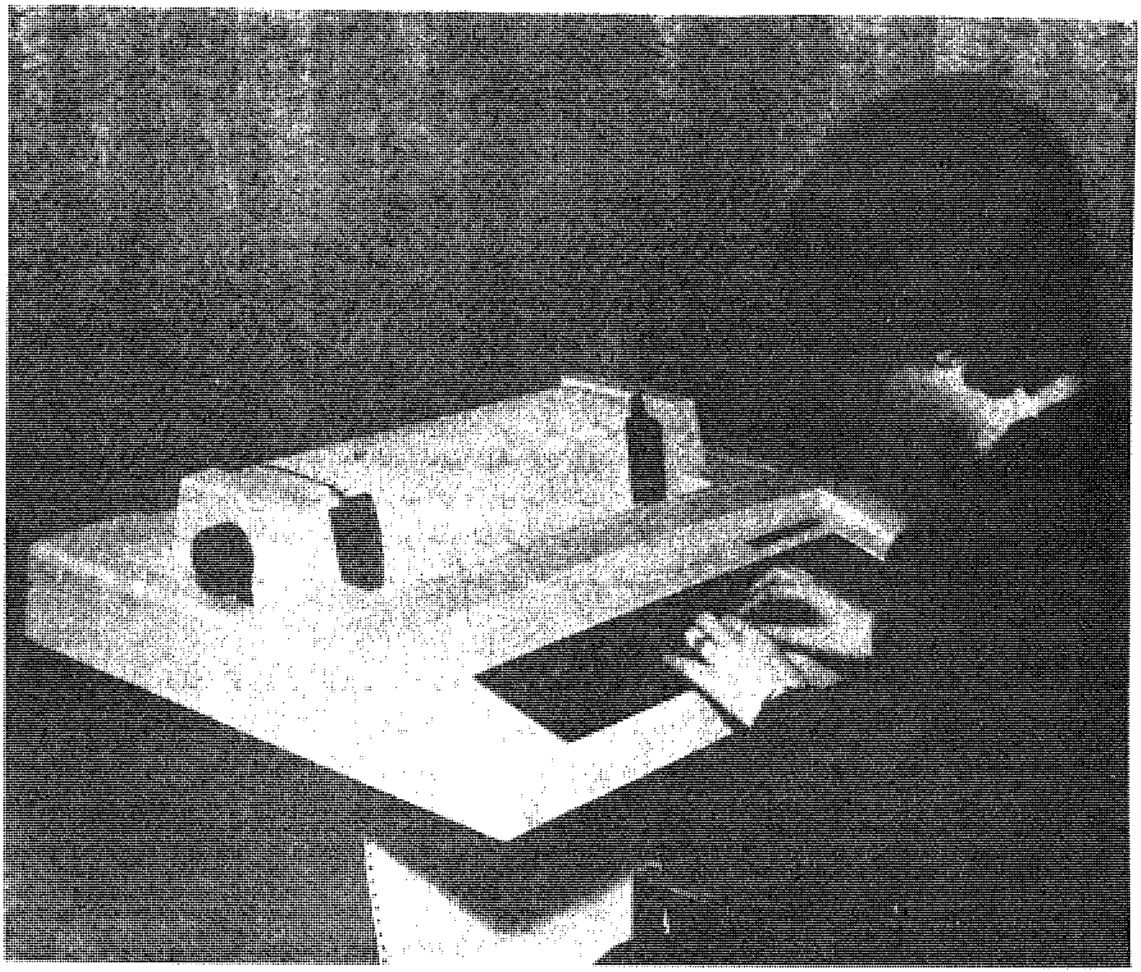
ويمكن نقل الكميات الكبيرة من المعلومات (البرامج والبيانات) إلى داخل الحاسب أو اخراجها بسرعة كبيرة فى التشغيل على دفعات . هذا بالإضافة إلى أنه لا يلزم المستخدم أن يتواجد أثناء العملية ، ولهذا فإن طريقة العمل هذه تناسب العمليات التى تحتاج كميات كبيرة من وقت الحاسب أو الطويلة جداً . ومن ناحية أخرى رغم أنه فى أى هذه العملية قد لا تحتاج لا إلى ثانية واحدة أو ثانيتين من الوقت الحقيقى للحاسب . (لابد للعملية من الانتظار حتى يأتى دورها وذلك قبل قراءتها وتشغيلها وطباعتها) ولهذا فمن الممكن أن يكون التشغيل على دفعات غير مرغوب فيه إذا كانت هناك ضرورة لتشغيل عدد كبير من العمليات البسيطة والصغيرة والحصول على نتائجها بأسرع ما يمكن .

يمكن إرسال كميات كبيرة من المعلومات (برامج وبيانات) من وإلى الحاسب بسرعة فائقة فى أسلوب التشغيل على دفعات . وهذا الأسلوب فى التشغيل يناسب الشغلات الطويلة أو التى تتطلب وقتاً طويلا من الحاسب . ومن الناحية الأخرى ، إجمالى الوقت المطلوب لتشغيل شغلة بهذه الطريقة يمكن أن يختلف من عدة دقائق إلى عدة ساعات رغماً عن أن الشغلة تتطلب فعلا ثانية أو اثنتين فقط من وقت الحاسب . (يجب أن تنتظر الشغلة دورها قبل أن تقرأ ثم تشغل وتطبع بعد ذلك) . وبذلك يكون التشغيل على دفعات غير مرغوب فيه عندما يكون المطلوب تشغيل عدة شغلات صغيرة وبسيطة وإرجاع النتائج بسرعة بقدر المستطاع .

الحساب التحاورى Interactive Computing

يمكن القيام بالحاسب التحاورى بحاسب شخصى صغير مثل المبين فى شكل ١ - ١ أو بنهاية طرفية لحاسب كما هو موضح فى شكل ١ - ٥ . وفى كلتا الحالتين يمد المستفيد الحاسب بالمدخلات من خلال لوحة مفاتيح وهى تشبه آلة كاتبة عادية . ثم تطبع النتائج المناظرة بعد ذلك إما على أفرخ كبيرة من الورق أو تعرض على وحدة TV . (يمكن أن تكون النتائج المطبوعة لازمة فى عدة أنواع من التطبيقات ، حيث أنها تعطى « نسخة ملموسة » (أو وثيقة مطبوعة) للجلسة التحاورية . إلا أن استخدام العرض على شاشة عرض TV غالباً ما يكون أكثر ملاءمة) . أحياناً ما يشارك إلى النهايات الطرفية التحاوريه للحاسبات .

الخاصية التى لها دلالة خاصة فى التشغيل التحاورى للحاسب هى أن المستفيد والحاسب قادران على التحدث كل مع الآخر أثناء جلسة الحسابات . وبذلك يمكن أن يطلب من المستفيد بصفة دورية أن يمد الحاسب بمعلومات معينة تقرر نوع الإجراءات التالية التى يقوم الحاسب بتنفيذها .



شكل ١ - ٣

مثال ١ - ٤

يرغب طالب فى استخدام حاسب شخصي لحساب نصف قطر دائرة مساحتها 100 . والمتاح هو برنامج يحسب مساحة الدائرة إذا أعطى نصف القطر . (لاحظ أن ذلك هو عكس الذى يرغبة الطالب تماماً) . وبالتالى فعلى الطالب أن يواصل بالمحاولة والخطأ ، تخمين قيمة لنصف القطر وتقرير ما إذا كانت النتيجة تناظر المساحة المطلوبة أم لا ، إن لم تكن فيفرض الطالب قيمة أخرى لنصف القطر ويحسب مساحة جديدة ، .. وهكذا .. وسوف يستمر فى طريقة المحاولة والخطأ حتى يجد الطالب قيمة لنصف القطر تعطى مساحة قريبة جداً من 100 .

وحالما يدار الحاسب ويتم إدخال البرنامج فسوف تطبع الرسالة :

RADIUS = ?

وبعد ذلك يدخل الطالب قيمة لنصف القطر . دعنا نفرض أن الطالب أدخل قيمة 5 لنصف القطر . بعد ذلك يستجيب الحاسب بطباعة :

AREA = 87.5398

DO YOU WISH TO REPEAT THE CALCULATIONS ?

ثم يطبع الطالب نعم أو لا . إذا طبع الطالب نعم فسوف تعاد طباعة الرسالة :

RADIUS = ?

ويعاد الإجراء بالكامل . أما إذا طبع الطالب لا فتطبع الرسالة :

GOODBYE

وتنتهي الحسابات :

```
RADIUS=? 5
AREA= 78.5398

DO YOU WISH TO REPEAT THE CALCULATION? YES

RADIUS=? 6
AREA= 113.097

DO YOU WISH TO REPEAT THE CALCULATION? YES

RADIUS=? 5.6
AREA= 98.5204

DO YOU WISH TO REPEAT THE CALCULATION? NO

GOODBYE
```

شكل ١ - ٤

نرى في شكل ١ - ٤ المعلومات التي تطبع خلال جلسة تحاورية نموذجية ، مستخدماً البرنامج الموصوف أعلاه . وقد تم وضع خط تحت المعلومات المطبوعة بواسطة الطالب . تم التوصل إلى قيمة تقريبية فيها $r = 5.6$ بعد ثلاثة حسابات .

لاحظ كيف تبدو طريقة تحدث الحاسب والطالب كل مع الآخر . ولاحظ أيضاً أن الطالب ينتظر حتى يرى مساحة محسوبة قبل تقرير ما إذا كان سيقوم بعمل الحسابات مرة أخرى أم لا . وعلى ذلك ، فإن القيمة الجديدة التي يعطيها الطالب لنصف القطر محكومة بتأثير النتيجة السابقة التي تم حسابها .

يقال أحياناً أن البرامج التي يتم تصميمها للتطبيقات من النوع التحاوري بأنها ذات طبيعة تخاطبية . الألعاب المبرمجة مثل أدفنشر — توونشيكرز والشطرنج كلها أمثلة ممتازة لمثل هذه التطبيقات التحاورية . وكذلك أيضاً الحركة السريعة للألعاب مثل غزاة الفضاء .

المشاركة الزمنية Timesharing

المشاركة الزمنية هي شكل من أشكال الحساب التحاوري حيث يستطيع عدد من المستفيدين استخدام حاسب واحد في نفس الوقت . يتصل كل مستفيد بالحاسب من خلال نهاية طرفية ، مثل المبينة في شكل ١ - ٥ . يمكن أن تتصل النهايات الطرفية بالحاسب بواسطة سلك ، أو يمكن أن تتصل بالحاسب من خلال خطوط تليفونية أو دائرة موجات دقيقة . وبذلك فإن النهاية الطرفية للمشاركة الزمنية يمكن وضعها بعيداً — ربما على بعد عدة مئات من الأميال عن الحاسب المضيف لها .

حيث أن الحاسب يعمل أسرع بكثير من شخص جالس أمام نهاية طرفية فإن حاسباً واحداً يمكن أن يخدم عدداً كبيراً من النهايات الطرفية في نفس الوقت . وبذلك لا يبالي أي مستفيد بوجود مستفيدين آخرين ، ويبدو له أن الحاسب بالكامل يعمل من أجله فقط وتحت أمره . وعادة ما تستخدم الحاسبات الدقيقة مثل التي في شكل ١ - ١ بدلاً من النهاية الطرفية للمشاركة الزمنية . فقد أصبحت الوصلات شائعة على وجه الأخص من خلال خطوط تليفونية . ويسمح هذا التنظيم لشخص يعمل من منزله على حاسب شخصي أن يتصل بحاسب بعيد في المدرسة أو المكتب .

وتلائم المشاركة الزمنية تشغيل الشغلات البسيطة نسبياً والتي لا تتطلب نقل كميات كبيرة من البيانات أو تستهلك وقتاً طويلاً من الحاسب . وهي نفس الخصائص التي تحملها معظم تطبيقات الحاسب التي تبرز في المدارس والمكاتب التجارية . ويمكن تشغيل هذه التطبيقات سريعاً وببساطة وبأقل التكاليف باستخدام المشاركة الزمنية .

مثال ١ – ٥

جامعة كبيرة بها حاسب وله إمكانيات المشاركة الزمنية ويتكون من 100 نهاية طرفية و 80 خط تليفون منفصل وتستخدم نظام المشاركة الزمنية وموضوعة فى أماكن مختلفة من مبنى الجامعة . وتتصل هذه النهايات الطرفية بالحاسب الكبير عن طريق خطوط تليفونية . ترسل كل من هذه النهايات البيانات من وإلى الحاسب بسرعة قصوى مقدارها 120 حرف/ثانية . ويمكن استخدام كل النهايات الطرفية فى آن واحد ، وبالرغم من ذلك فإنها تتفاعل مع حاسب واحد فقط . وتسمح الخطوط التليفونية للطلبة الغير موجودين بمبنى الجامعة بتوصيل حاسباتهم الشخصية بالحاسب الكبير ويمكن لكل حاسب شخصى أن ينقل بيانات من وإلى الحاسب الكبير بسرعة قصوى قدرها 120 حرف/ثانية وعلى ذلك فإنه يمكن للنهايات والحاسبات كلها وعددها 180 أن تتحاور مع الحاسب الرئيسى فى وقت واحد . ومع كل فإن كل طالب لا يعلم أن الآخرين يستخدمون الحاسب فى نفس الوقت .

وبالإضافة لذلك ، تم توصيل 20 نهاية طرفية عن بعد للحاسب . وضعت 15 نهاية طرفية منها فى 5 مدارس ثانوية فى المنطقة ، والنهايات الخمس الباقية وضعت فى معمل أبحاث حكومى . جميع النهايات الطرفية وعددها 120 فى آن واحد (وكثيراً ما يحدث ذلك) . وباقتسام الحاسب بهذه الطريقة يستطيع كل معهد أن ينتفع بخدمات الحاسب الكبير بتكلفة مناسبة .

يمكن – إذا تطلب الأمر ذلك – إدماج خصائص معينة من أسلوب التشغيل على دفعات وأسلوب المشاركة الزمنية ، فمثلا ، يمكن إدخال مجموعة بيانات مباشرة من النهاية الطرفية (وبذلك نكون فى غنى عن التثقيب) ثم بعد ذلك يستكمل التشغيل بطريقة وأسلوب التشغيل على دفعات . ويمكن أيضاً استخدام قارئ البطاقات (تشغيل على دفعات) لإدخال برنامج ومجموعة بيانات ، ثم ينقح (يعدل) البرنامج وتشغل البيانات بأسلوب المشاركة الزمنية . مثل هذه العمليات الخليطة أصبحت أكثر شيوعاً كلما ازدادت أنظمة الحاسبات تعقيداً .

١ – ٤ أنواع لغـــات البرمجة TYPES OF PROGRAMMING LANGUAGES

يمكن استخدام عدة لغات مختلفة لبرمجة الحاسب . واللغة الأساسية هى **لغة الآلة** – وهى مجموعة تفصيلية من التعليمات المكودة والتى تتحكم فى دوائر الحاسب الداخلية . وهذه هى اللهجة الطبيعية للحاسب . وقد تمت كتابة مجموعة قليلة من برامج الحاسب فعلا بلغة الآلة ، ومع ذلك ، ولسببين هامين : أولهما ، أن لغة الآلة مرهقة جداً للعمل بها ، وثانيهما ، أن لكل حاسب مجموعة الأوامر الفريدة الخاصة به . (وبذلك فإن البرنامج المكتوب بلغة الآلة لنوع معين من الحاسبات لا يمكن تشغيله على نوع آخر من الحاسبات بدون تعديلات جوهرية) .

وعادة ، تم كتابة برامج الحاسب ببعض **اللغات العالية المستوى** حيث تتفق مجموعة الأوامر الخاصة بها مع لغات وأفكار الإنسان . معظم هذه اللغات العالية المستوى لغات **لأغراض عامة** مثل البيسك (بعض اللغات الأخرى الشائعة الاستخدام ولأغراض عامة أخرى هى (Fortran, Cobol ، Pascal و PL/I) . ويوجد أيضاً العديد من اللغات العالية المستوى **لأغراض خاصة** حيث تصمم مجموعة الأوامر الخاصة بها لأنواع معينة من التطبيقات .

وكقاعدة فإن الأمر الواحد فى لغة عالية المستوى يكون مساوياً – لعدة أوامر من لغة الآلة . علاوة على ذلك ، فإن البرنامج المكتوب بلغة عالية المستوى يمكن تشغيله بصفة عامة على عدة أنواع مختلفة من الحاسبات بقليل من التعديلات أو بدون تعديلات على الإطلاق . ومعنى ذلك فإن اللغة العالية المستوى تقدم لنا بعض المزايا الهامة عن استخدام لغة-الآلة ألا وهى **البساطة والتناسق والقابلية للنقل** (أى الاستقلال عن الآلة) .

وبذلك فإن البرنامج المكتوب بلغة عالية المستوى يجب أن يترجم إلى لغة الآلة قبل أن ينفذ . وتعرف هذه المرحلة **بالتفسير أو الترجمة** ، معتمداً فى ذلك على الطريقة المعمول بها . (معظم نسخ البيسك يتم تفسيرها وليس ترجمتها) . البرامج المفسرة أسهل فى العمل بها عن البرامج المترجمة بالرغم من أن البرامج المترجمة تنفذ أسرع بصفة عامة . ومع ذلك فى كلتا الحالتين تنفذ

الترجمة أتوماتيكياً بداخل الحاسب . وفى الحقيقة فإن المبرمج عديم الخبرة ربما لا يعى أن هذا الإجراء يتم ، حيث أنه يرى فقط البرنامج الأصلى وبيانات الإدخال والنتائج وبيانات الإخراج .

المفسر والمترجم ما هو إلا برنامج **الحاسب** يقبل أى برنامج مكتوب بلغة عالية المستوى كبيانات إدخال ويولد برنامج مناظر مكتوب بلغة الآلة كمخرج . وتبعاً لذلك ، فإن البرنامج الأصلى المكتوب بلغة عالية المستوى يسمى برنامج المنبع ، والبرنامج الناتج المكتوب بلغة الآلة يسمى برنامج **الهدف** . ويجب أن يكون لكل حاسب مفسر أو مترجم لكل لغة عالية المستوى خاص بها . واستخدام المفسرات والمترجمات هو الذى يمكننا من الحصول على التناسق والاستقلال عن الآلة مع لغة عالية المستوى مثل لغة البيسك .

١ — ٥ مقدمــة للغــة البيسك INTRODUCTION TO BASIC

إن لغة البيسك سهلة فى الاستخدام « مألوفة » تجمع أوامرها ما بين الصيغ الجبرية البسيطة وبعض الكلمات الدالة باللغة الإنجليزية مثل LET و READ و PRINT و GO TO و IF و THEN ، ... إلخ . معظم اللغات العالية المستوى لها هياكل متشابهة ولكنها أصعب فى التعلم والاستخدام عن لغة البيسك . وبذلك ، فإن البيسك تناسب بصفة عامة الأشخاص الذين يتعلمون البرمجة لأول مرة وفى الحقيقة فإن كثيراً من المدارس الثانوية والمدارس الإعدادية يعرض الآن دروساً فى البرمجة بلغة البيسك ، كما أن عدداً من المدارس الأولية تقدم الآن الموضوع بصورة اختيارية لمجموعات من الطلبة .

إن استخدام البيسك مفيد بدون شك وغير مقصور فى استخدامه للتطبيقات الأولية . ويستخدم غالباً فى عديد من التطبيقات المتقدمة المتنوعة فى مجالات معينة مثل التجارة والاقتصاد وعلم النفس والطب وأيضاً فى المجالات العلمية والهندسية والرياضيات . ولقد أصبح البيسك أيضاً اللغة الأولى « لهواة » الحاسبات الدقيقة الذين تتضمن اهتماماتهم ألعاب الحاسب التى تحتاج إلى استخدام الرسوم وتوليد الأصوات وكذلك التطبيقات التقليدية مثل الحسابات الشخصية وقواعد البيانات الإدارية .. إلخ وسوف نرى عينات مختارة لعدة أنواع مختلفة من التطبيقات فى الأمثلة المبرمجة المتضمنة فى هذا الكتاب .

نبذة تاريخية عن البيسك History of BASIC

تطور البيسك أساساً فى كلية دارتموث الأمريكية بواسطة جون كيمنى وتوماس كورتز فى منتصف الستينات . وقد تم اكتشافه سريعاً وتبناه عدد من تجار خدمات المشاركة الزمنية مما ساعد فى انتشار اللغة وإلقاء الضوء عليها بين آلاف من مستخدمى الحاسبات . وبسرعة عرضت معظم الشركات المنتجة للحاسبات نسخ البيسك الخاصة بحاسباتهم . وبذلك أصبحت لغة البيسك بسرعة لغة المشاركة الزمنية ومن أكثر اللغات شيوعاً واستخداماً فى الولايات المتحدة .

تلقى البيسك دفعة هامة مع تطور الحاسبات الدقيقة المنخفضة التكاليف التى بدأت فى منتصف السبعينات . وحقيقة فإن كل الحاسبات الدقيقة قد تبنت البيسك كلغة برمجة أساسية لها ، كما أن عدداً من الحاسبات الدقيقة قد ضمنت مفسراً للغة البيسك كجزء دائم فى دوائرها الداخلية . فى الحقيقة ، فإن بساطة وملاءمة البيسك كان عاملاً أساسياً فى التكاثر السريع لهذه الأجهزة . وأصبحت لغة البيسك متاحة فى كل حاسب دقيق مباع ، وهناك العديد من الحاسبات الدقيقة تحتوى تصميم مفسر بيسك كجزء فى دائرتها الداخلية .

فى عام 1978 وضع المعهد القومى الأمريكى للمعايرة (ANSI) تقنينات لمجموعة فرعية مهمة من البيسك* ، من أجل زيادة التناسق بين نسخ البيسك وبعضها . ومع ذلك ، فإن معظم نسخ البيسك تحتوى على بعض الخصائص التى لا يتضمنها ANSI القياسى . وبذلك فيوجد بعض التغيرات فى نسخ البيسك المختلفة المتاحة حالياً . وجار تطوير نسخة أكثر شمولا من ANSI القياسى والتى سوف تتضمن هذه الخصائص الإضافية .

---

* American National Standard to Minimal BASIC, ANSI X 3.6 – 1978, American National Standards Institute, New York 1978.

التغيرات فى البيسك Variations in BASIC

معظم نسخ البيسك المنفذة على الحاسبات الكبيرة أو المدعمة بواسطة خدمات المشاركة الزمنية يشبه بعضها الآخر . وكلها تتضمن الخواص الموضحة فى النسخة المعيارية ANSI لعام ١٩٧٨ ، علاوة على عدد إضافى من الخواص الشائعة المستخدمة . وغالباً ما يشار إلى مثل هذه التفاوتات بتفاوتات البيسك الخاص بدارتموث . ويمكن لمعظم البرامج التى تنتفع بخواص البيسك الخاص بدارتموث أن تشغل على بعض الحاسبات المختلفة بتعديلات بسيطة أو بدون تعديلات على الإطلاق .

وقد تسببت المنافسة بين منتجى الحاسبات لإحباط أى محاولة فى التقنيات الرسمية ، ويتم تعزيز معظم نسخ البيسك للحاسب الدقيق بأوامر خاصة تتفق مع امكانيات الأجهزة الخاصة بها . وعلى ذلك فإن هناك أوامر خاصة للقيام بإجراء الرسومات ، لتوليد الصوت والتحكم فى الآلات المساعدة للحاسبات الدقيقة مثل الأقراص المرنة وآلات الطباعة والأقلام الضوئية وعصا التوجيه . وتتشابه هذه الأوامر من نسخة بيسك حاسب دقيق إلى أخرى وعلى ذلك فإن العديد من برامج البيسك التى تكتب لأحد الحاسبات الدقيقة يمكن تعديلها لتعمل على حاسب دقيق آخر دون جهد كبير .

ولابد من اتباع هذه القواعد الست فى أمثلة البرامج التى تعرض فى أول ثمانية فصول فى هذا الكتاب ، وعندما نصل إلى الفصل التاسع سنجد أنه يمكن التراخى عنها فى بعض نسخ بيسك الحاسبات الدقيقة ، ومع كل فإن البرامج التى تطبق هذه القواعد ستظل صحيحة .

وقد تم فى السنوات الأخيرة تطوير نسخ أكثر حداثة من البيسك تتضمن بعض الخصائص المعقدة المختلفة والتى لا توجد فى النسخ التقليدية . ومن المدهش أن معظم هذه النسخ المعدلة من اللغة قد تم تطويرها للحاسبات الدقيقة وأن كان بعض هذه النسخ المعدلة من البيسك قد تم تطويرها أيضاً للحاسبات الكبيرة .

يصف هذا الكتاب كل الخصائص الشائعة فى البيسك . ويتضمن العديد من الخصائص الأكثر حداثة والمعدلة بجانب الخصائص المعيارية الموجودة فى العديد من النسخ التقليدية للغة . وبالتحديد ، فإن المواد الموجودة فى الفصول من الثانى إلى السادس تطبق بانتظام على كل نسخ بيسك .

تتضمن أيضاً معظم نسخ البيسك للحاسبات الكبيرة الخواص الموضحة فى الفصل السابع (المتجهات والمصفوفات) . وتطبق المواد الموجودة فى الفصل الثامن (ملفات البيانات) على أعداد كبيرة من نسخ البيسك ، بالرغم من وجود اختلافات يجب أخذها فى الاعتبار بالطريقة الصحيحة عند تنفيذ هذه الخواص . وأخيراً تصف الفصول من التاسع إلى الثانى عشر العديد من الخصائص الخاصة والطرق الفنية الموجودة فى بيسك الحاسبات الدقيقة مع التأكيد الخاص على ميكروسوفت بيسك المنفذ على حاسب IMB الشخصى ويحتوى الكتاب أيضاً على خمس ملاحق والتى تلخص معظم المادة التى سبق عرضها.

ولن يواجه القارئ المتمكن من المادة العلمية أى صعوبة فى استغلال الخواص المتاحة فى نسخة معينة من البيسك ، أو فى تغيير برنامج حتى يمكن تشغيله مع نسخة مختلفة من اللغة .

بناء برنامج البيسك Structure of a BASIC Program

يكتب كل أمر فى برنامج البيسك كجملة منفصلة . وبذلك فإن برنامج البيسك الكامل سوف يتكون من جمل متتالية . يجب أن تظهر هذه الجمل بالترتيب الذى يجب أن تنفذ به إلا إذا تمت الإشارة « للقفز » عن قصد (أى تحويل التحكم) .

تطبق القواعد التالية على جمل البيسك :

١ - يجب ظهور كل جملة على سطر منفصل(*) .

٢ - لا يمكن أن تتعدى الجملة طول سطر واحد (أى لا يمكن « استكمالها » من سطر لآخر) .

---

* على معظم النهايات الطرفية لحاسب ، السطر يساوى 80 عموداً ؛ ومع ذلك ، فبعض النهايات الطرفية تسمح بعدد من الحروف يصل إلى 132 حرفاً / سطر .

٣ – يجب أن تبدأ كل جملة بكمية صحيحة موجبة تعرف كرقم الجملة (أو رقم السطر) .

٤ – يجب أن تكون للجمل المتعاقبة أرقام سطور متزايدة .

٥ – يجب أن يتبع كل رقم سطر كلمة دالة من كلمات البيسك تشير لنوع الأمر الذى يجب القيام به .

٦ – يمكن إضافة الفراغات فى أى مكان مطلوب من أجل تحسين قراءة الجملة .

يمكن أيضاً تضمين السطور الفارغة (الخالية) فى برنامج البيسك . مع ذلك ، فإن كل سطر خال يجب أن يحمل رقم سطر فريد ، ويتبع رقم السطر مسافة خالية واحدة على الأقل . (أما إذا تلى رقم السطر رجوع عربة الآلة فوراً . فإن رقم السطر هذا سوف يلغى من البرنامج . وسوف يتم مناقشة ذلك فى الفصل الثالث) .

ولابد من اتباع هذه القواعد الست فى أمثلة البرامج التى تعرض فى أول ثمانية فصول من هذا الكتاب ، وعندما نصل إلى الفصل التاسع سنجد أنه يمكن التراخى عنها فى بعض نسخ بيسك الحاسب الدقيق ، ومع كل فإن البرامج التى تطبق هذه القواعد ستظل صحيحة .

مثال ١ – ٦ مساحة دائرة

يمثل شكل ١ – ٥ برنامج بيسك بسيط لحساب مساحة دائرة معروف نصف قطرها . والمنطق المستخدم لإجراء هذه الحسابات قد تم مناقشته فى مثال ١ – ١ . إن البرنامج بسيط للغاية ، ومع ذلك فإن الأساس المنطقى يمكن تحديده بواسطة عملية فحص بسيط .

```
10 INPUT R
20 LET A=3.14159*R^2
30 PRINT R,A
40 END
```

شكل ١ – ٥

نرى أن البرنامج يتكون من أربع جمل كل منها يظهر فى سطر منفصل . كل جملة لها رقم جملة خاص بها (أو رقم سطر) . وتزيد هذه الأرقام بالتعاقب من بداية (أعلى) إلى نهاية (أسفل) البرنامج . تحتوى الجمل على كلمات بيسك الدالة وهى END و PRINT و LET و INPUT بالترتيب

الغرض من الجملة الأولى (INPUT R) هو إدخال قيمة رقمية لنصف القطر R من لوحة التشغيل المركزية . وتسبب الجملة الثانية (LET A = 3.14159*R ↑ 2) فى إيجاد قيمة R^2 . ثم بعد ذلك سوف تمثل هذه الكمية بالحرف A (للمساحة) . وتسبب الجملة الثانية (PRINT R,A) فى إرسال القيم الرقمية R و A، للوحة التشغيل المركزية ، حيث يتم طباعتها بعد ذلك . وأخيراً ، فإن آخر جملة (END) مطلوبة للتعبير عن انهاء البرنامج .

لاحظ الرموز المستخدمة فى السطر 20 للتعبير عن العمليات الحسابية . فيشار إلى الضرب بواسطة نجمة (*) ، ويستخدم السهم الرأسى (↑) لرفع كمية لقوة معينة . (هذه العملية الأخيرة معروفه كعملية الأس وتستخدم بعض الطرفيات علامة « ^ » للتعبير عن الأس كما هو مبين بشكل ١ – ٥) . وتمثل العمليات الحسابية الأخرى والمسماة بالجمع والطرح والقسمة فى البيسك بواسطة علامة الجمع (+) وعلامة الطرح (—) والخط المائل (/) على الترتيب .

بعض مزايا لغة البيسك Some Advantages of BASIC

١ – إن لغة البيسك لغة « صديقة » أى « مناسبة للإنسان » . فهى سهلة للتعلم ومسلية فى الاستخدام . أى شخص منظم يمكنه تعلم كيف يبرمج بلغة البيسك وليس من الضرورى أن يكون عنده خلفية كبيرة فى الرياضيات .

٢ – إن اللغة غاية فى المرونة ، وتسمح للمبرمج بتطوير برامج جديدة والتغيير فى البرامج الموجودة بمجهود قليل نسبياً .

٣ – تناسب لغة البيسك استخدام البيئة التحاورية . وهذا يتضمن التطبيقات المخصصة للحاسبات الدقيقة بجانب تطبيقات المشاركة الزمنية للحاسبات الكبيرة .

٤ – إن اللغة متاحة عالمياً على كل من الحاسبات الكبيرة والصغيرة . وقد أصبحت لغة البرمجة القياسية لمعظم تطبيقات الحاسبات الدقيقة .

٥ – إن الخواص الشائعة الاستخدام فى البيسك قياسية نسبياً ، بالرغم من احتمال وجود فروق بسيطة بين نسخة من البيسك وأخرى . وبذلك فإن اللغة مستقلة عن الآلة استقلالا كبيراً . وبالتالى ، فإن معظم برامج البيسك يمكن تشغيلها على حاسبات مختلفة كثيرة بقليل من التعديلات أو بدون تعديلات إطلاقاً .

اسئلة للمراجعة
Review Questions

١ – ١ ما هو المقصود بحاسب كبير ؟ وأين توجد الحاسبات الكبيرة وفيما تستخدم عادة ؟

١ – ٢ ما هو الحاسب الصغير ؟ وكيف يختلف الحاسب الصغير عن الحاسب الكبير ؟

١ – ٣ ما هو الحاسب الدقيق ؟ وكيف تختلف الحاسبات الدقيقة عن الحاسبات الكبيرة والحاسبات الصغيرة .

١ – ٤ اذكر نوعين مختلفين من البيانات .

١ – ٥ ما هو المقصود ببرنامج الحاسب ؟ وماذا يحدث ، وعموماً ، عند تنفيذ برنامج الحاسب ؟

١ – ٦ ما هى ذاكرة الحاسب ؟ ما هى أنواع المعلومات التى تخزن فى ذاكرة الحاسب ؟

١ – ٧ ما هو الرقم الثنائى ؟ ما هو البايت ؟ ما هو الفرق بين البايت والكلمة فى الذاكرة ؟

١ – ٨ ما هو المصطلح المستخدم لوصف حجم ذاكرة الحاسب ؟ ما هى بعض الأحجام النموذجية للذاكرة ؟

١ – ٩ اذكر أسماء بعض الأجهزة النموذجية للذاكرة المساعدة . وكيف يختلف هذا النوع من الذاكرة عن الذاكرة الأساسية للحاسب ؟

١ – ١٠ ما هى وحدة قياس الزمن المستخدمة للتعبير عن سرعة المهام الأساسية التى يؤديها الحاسب ؟

١ – ١١ ما هو الفرق بين أسلوب التشغيل على دفعات وأسلوب التشغيل التحاورى ؟ ما هى ميزاتهما وعيوبهما ؟

١ – ١٢ ما هو المقصود بالمشاركة الزمنية ؟ ما هى أنواع التطبيقات التى تناسبها المشاركة الزمنية ؟

١ – ١٣ ما هى لغة الآلة ؟ وكيف تختلف عن لغات المستوى العالى ؟

١ – ١٤ اذكر بعض لغات مستوى عال شائعة الاستخدام . وما هى مميزات استخدام لغات المستوى العالى ؟

١ – ١٥ ما هو المقصود بالتفسير أو الترجمة وكيف تختلف العمليتان ؟

١ – ١٦ ما هو برنامج المنبع ، برنامج الهدف ، ولماذا تكون هذه المفاهيم مهمة ؟

١ – ١٧ ماذا تعنى كلمة البيسك ؟

١ – ١٨ ما هى الخواص العامة للغة البيسك ؟

١ – ١٩ أين تم وضع لغة البيسك ؟ ومن هو الذى قام بذلك ؟

١ – ٢٠ إلى أى مدى تختلف كل نسخة مستقلة من البيسك عن أخرى ؟ وهل تم توحيد معيارى للغة ؟

١ – ٢١ ما هى جملة البيسك وبأى ترتيب يجب ظهور الجمل فى برنامج البيسك ؟

١ – ٢٢ لخص القواعد الست التى تطبق على كل جمل البيسك . وهل هذه القواعد تطبق فى معظم نسخ بيسك الحاسب الدقيق .

١ – ٢٣ ما هى الرموز التى تستخدم فى البيسك لتشير إلى الجمع والطرح والضرب والقسمة ؟

١ – ٢٤ ما هو المقصود بالأس ؟ وما هو الرمز المستخدم فى البيسك لتمثيل الأس ؟

١ – ٢٥ لخص المزايا الأساسية للغة البيسك .

مســائل محلولة

Solved Problems

١ – ٢٦ عدة برامج بيسك أولية معروضة فيما يلى . اشرح غرض كل برنامج منها

(أ)

```
10 INPUT L,W
20 LET A=L*W
30 PRINT L,W,A
40 END
```

يحسب هذا البرنامج مساحة مستطيل معروف طوله وعرضه .

(ب)

```
10 INPUT A,B,C,D,E
20 LET S=A+B+C+D+E
30 PRINT A,B,C,D,E
40 PRINT S
50 END
```

يحسب هذا البرنامج حاصل جمع 5 اعداد . لاحظ أن الأعداد الخمسة سوف تطبع على سطر واحد ويطبع حاصل الجمع المحسوب على السطر التالى . (كل جملة من جمل الطباعة PRINT تبدأ من سطر جديد) .

(جـ)

```
10 INPUT A,B,C
20 LET X1=(-B+(B↑2-4*A*C)↑.5)/(2*A)
30 LET X2=(-B-(B↑2-4*A*C)↑.5)/(2*A)
40 PRINT A,B,C,X1,X2
50 END
```

يحسب هذا البرنامج قيم x_1 و x_2 من الصيغ الرياضية

$$x_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

$$x_2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

حيث تحدد قيم a و b و c

١ – ٢٧ اكتب برنامج بيسك أولى لكل من الحالات الموصوفة فيما يلي :

(أ) احسب نصف قطر دائرة معروفة المساحة (انظر مثال ١ – ٧).

بما أن $A = \pi r^2$ فيمكننا حلها لإيجاد قيمة r ، بحيث تكون

$$r = \sqrt{A/\pi}$$

وبذلك فإن البرنامج المطلوب هو :

```
10 INPUT A
20 LET R=(A/3.141593)↑.5
30 PRINT A,R
40 END
```

(ب) احسب طول مستطيل معروف المساحة والعرض (انظر المسألة (١ – ٢٦ أ).

```
10 INPUT A,W
20 LET L=A/W
30 PRINT A,W,L
40 END
```

(جـ) احسب حاصل ضرب 5 أرقام معطاة (انظر مسألة ١ – ٢٦ ب).

```
10 INPUT A,B,C,D,E
20 LET P=A*B*C*D*E
30 PRINT A,B,C,D,E
40 PRINT P
50 END
```

مسائل تكميلية (اضافية)

Supplementary Problems

١ – ٢٨ نعرض فيما يلي عدة برامج أساسية للبيسك . اشرح الغرض من كل برنامج .

(ا)

```
10 INPUT B,H
20 LET A=(B*H)/2
30 PRINT B,H,A
40 END
```

(ب)

```
10 INPUT L,W,
20 LET C=2*(L+W)
30 PRINT L,W,C
40 END
```

(جـ)

```
10 INPUT U,V
20 LET W=U+V
30 LET X=U−V
40 LET Y=U*V
50 LET Z=U/V
60 PRINT U,V
70 PRINT W,X,Y,Z
80 END
```

(د)

```
10 INPUT X
20 LET Y=1+X+(X↑2)/2+(X↑3)/6
30 PRINT X,Y
40 END
```

١ – ٢٩ اكتب برنامج بيسك أولى لكل من الحالات الموصوفة فيما يلي :

(ا) احسب محيط دائرة إذا أعطى نصف قطرها (انظر مثال ١ – ٧) .

(ب) احسب طول وتر المثلث القائم الزاوية إذا أعطيت كلا من القاعدة والارتفاع (انظر المسألة (١ – ٢٨ أ)) .

(جـ) احسب قيمة الصيغة الرياضية

$$w = \frac{u - v}{u + v}$$

حيث تعطى قيم u ، v (انظر المسألة (١ – ٢٨ جـ)) .

(د) احسب قيمة الصيغة الرياضية :

$$y = 100(1 + x + 2x^2 + 3x^3)$$

حيث توصف قيمة x (انظر المسألة (١ - ٢٨ د)).

١ - ٣٠ فيما يلي برنامج بيسك لحساب مساحة ومحيط مستطيل وكذا طول القطر . ولكن بعض الجمل مكتوبة بطريقة غير صحيحة . حدد جميع الأخطاء .

```
10 INPUT L,W
20 LET A=L*W        30 LET P=2*(L+W)
35 D=(L↑2+W↑2)↑.5
25 PRINT L,W,A,P,D
40 END
```

الفصل ٢

طريق البداية للغة البيسك

Getting Started with BASIC

سوف نفحص فى هذا الفصل عدة مفاهيم جوهرية للغة البيسك ، مثل الأرقام ، والمتغيرات والصيغ الرياضية . ثم بعد ذلك سوف ندرس الست جمل الأكثر شيوعاً عند استخدام البيسك ، والتى تسمح لنا بعمل إدخال / إخراج وعملية تداول البيانات والقفز إلى أجزاء أخرى من البرنامج عندما نرغب ذلك . وبعد الانتهاء من هذا الفصل سوف يكون القارىء قادراً على كتابة برامج بيسك خاصة به لمسائل مختلفة .

٢ ــ ١ الأرقــام (الثوابت) NUMBERS (CONSTANTS)

يشار للكميات الرقمية فى البيسك كأرقام (أو ثوابت) ويمكن التعبير عن الأرقام بطريقتين مختلفتين : كميات صحيحة (أرقام صحيحة بدون علامة عشرية) أو كميات عشرية (أرقام بها علامة عشرية) . تطبق القواعد الآتية على كتابة الأرقام :

١ – لايمكن للفصلة أن تظهر فى أى مكان من الرقم .

٢ – يمكن أن يسبق الرقم علامة + أو – (ويفهم أن الرقم موجب إذا لم تظهر أى علامة) .

٣ – يمكن أن يحتوى الرقم على أس ، إذا نطلب ذلك . يشبه التمثيل الأسى التمثيل العلمى ، ماعدا أن الأساس 10 يستبدل بالحرف E وبذلك فإن الكمية 1.2×10^{-3} يمكن كتابتها فى البيسك 3 — 1.2 E ويمكن للأس أن يكون إما موجباً أو سالباً ولكن لا يمكن أن يحتوى على علامة عشرية .

٤ – معظم نسخ البيسك تسمح أن يصل الرقم إلى 8 أو 9 خانات معنوية .

٥ – يمكن أن تكون القيمة كبيرة لتصل إلى 10^{38} أو صغيرة لتصل إلى 10^{-38} (وتختلف هذه الكمية من نسخة بيسك لأخرى) ويسمح أيضاً بالصفر .

مثال ٢ – ١

تعبر الكميات العددية التالية عن أرقام مقبولة فى البيسك . لاحظ أن كل كمية (كل صف) يمكن أن تكتب بعدة طرق مختلفة .

0	+0	−0
1	+1	0.1E+1
−5280	−5.280E+3	−.5280E4
+1492	1492	1.492E+3
−.0000613	−6.13 E−5	−613 E−7
3000000	3E6	3 E+6

٢ ــ ٢ سلاسل الحروف STRINGS

سلسلة الحروف هى عدة حروف متتالية (أى حروف ، أعداد ، حروف خاصة معينة مثل + و – و / و * و = و $ و ... الخ) . يمكن أن تتضمن سلسلة الحروف أماكن خالية ولكن لايمكن أن تتضمن علامات الاقتباس . وأقصى عدد يمكن أن تتضمنه سلسلة الحروف يختلف من نسخة بيسك لأخرى . فى بعض النسخ لايمكن أن تتعدى سلسلة الحروف 15 حرفاً ، بينما تسمح نسخ أخرى بعدد يصل إلى 4095 حرفاً .

تستخدم سلاسل الحروف لتمثل المعلومات غير الرقمية مثل أسماء وعناوين و . . . الخ . وهى تستخدم أيضاً لعنونة بيانات المخرجات العددية ولطباعة الرسائل النصية (التطبيقات التى سوف نراها فى الكتاب بعد ذلك) .

مثال ٢ - ٢

فيما يلى عدة سلاسل الحروف

SANTA CLAUS	TYPE A VALUE FOR C:
APOLLO-17	$19.95
X1=	3730425
DO YOU WISH TO TRY AGAIN?	THE ANSWER IS

لاحظ أن الأرقام الصحيحة المتتالية مثل 3730425 لا تمثل كمية عددية إذا كتبت كسلسلة من الحروف .

٢ - ٣ المتغيرات VARIABLES

المتغير هو اسم يمثل رقماً أو سلسلة حروف . وفى نسخ بيسك القديمة كل متغير عددى يجب أن يتكون من حرف أو حرف يتبعه رقم صحيح . **المتغير الحرفى** يجب أن يكتب كحرف يتبعه علامة الدولار ($) . وتسمح أيضاً معظم نسخ البيسك للمتغير الحرفى أن يكتب فى صورة حرف يتبعه رقم صحيح ويتبعه علامة الدولار $.

مثال ٢ - ٣

يمكن أن تمثل كل من المتغيرات التالية كميات عددية :

A K X C1 X5

يمكن أن تمثل كل من المتغيرات التالية سلسلة من الحروف :

A$ K$ X$ C$ T$

فى معظم نسخ البيسك يمكن أن تمثل سلسلة الحروف بواسطة أى من المتغيرات التالية :

A2$ K9$ X0$ C1$ X5$

٢ - ٤ المعاملات والصيغ الرياضية (التعبيرات الرياضية)

OPERATORS AND FORMULAS (EXPRESSIONS)

تستخدم فى البيسك رموزاً خاصة تسمى **معاملات** وذلك لتشير للعمليات الحسابية الجمع والطرح والضرب والقسمة والأس . هذه المعاملات هى :

الجمع	+	(علامة الجمع)
الطرح	−	(علامة الطرح)
الضرب	*	(نجمـــــة)
القسمة	/	(شرطة مائلة)
الأس	↑	(سهم رأسى يشير إلى أعلى)

(بعض النهايات الطرفية تستخدم الرمز المبيّن)

تستخدم المعاملات لتصل بين الأرقام والمتغيرات العددية ، وبذلك تكون **صيغاً رياضية (أو تعبيرات رياضية)** .

وتجرى العمليات المشار إليها على الحدود الرقية فى الصيغ الرياضية وتنتج قيمة عددية وحيدة . ومن ثم فإن الصيغ الرياضية تمثل كية رقية محدودة .

مثال ٢ – ٤

نعرض فيما يلى عدة صيغ رياضية للغة بيسك :

```
J+1
A+B−C
(2*X−3*Y)/(U+V)
3.141593*R↑2
B↑2−4*A*C
```

تمثل كل صيغة كية رقية . وبذلك إذا كانت المتغيرات A و B و C تمثل الكيات الرقية 2 و 5 و 3 على الترتيب فسوف تتمثل الصيغة الرياضية A + B — C بالكية 4 .

وبالتحديد ، فإن الصيغة الرياضية يمكن أن تتكون من عدد وحيد أو متغير رقى وحيد ، أو توافقية من الأعداد والمتغيرات العددية والمعاملات . ومع ذلك ، فن المهم أن تفهم أن المتغير العددى يجب أن تحدد قيمته بكية رقية قبل أن يظهر فى الصيغة الرياضية . وإلا فلا يمكن حساب قيمة الصيغة الرياضية لتنتج قيمة عددية .

٢ ـــ ٥ التدرج الهـــرمى للعمليات الحسابية HIERARCHY OF OPERATIONS

إذا كان هناك معاملان أو أكثر فى أى صيغة رياضية ، يمكن أن تبرز أسئلة بمعنى هل تناظر الصيغة الرياضية 2 * X - 3 * Y الحد الجبرى $(2x) - (3y)$ أو $2(x - 3y)$ ؟ وبالمثل هل تناظر A/B * C لكية $a/(bc)$ أو $(a/b)C$ ؟ يمكن أن يجاب عن هذه الأسئلة بسهولة إذا عرفنا التدرج الهرمى للعمليات الحسابية ومرتبة التنفيذ بداخل مجموعة متدرجة .

التدرج الهرمى للعمليات الحسابية هو :

١ – **الأس** تؤدى كل عمليات الأس أولا .

٢ – **الضرب والقسمة** تجرى هذه العمليات الحسابية بعد أن ننتهى من تأدية كل العمليات الأسية . وليس من الضرورى أن يسبق الضرب القسمة .

٣ – **الجمع والطرح** هذه آخر العمليات التى نقوم بها . وليس من الضرورى أن يسبق الجمع الطرح .

تجرى هذه العمليات الحسابية بداخل مجموعة متدرجة من اليسار إلى اليمين .

مثال ٢ – ٥

الصيغة الرياضية :

```
A/B*C
```

مساوية للتعبير الرياضى $(a / b)c$ ، حيث أننا نقوم بالعمليات الحسابية من اليسار إلى اليمين .

وبالمثل الصيغة الرياضية :

```
B↑2−4*A*C
```

مساوية للتعبير الرياضى $b^2 - (4ac)$ فى هذه الحالة تجرى B ↑ 2 أولا ثم يتبعها حاصل الضرب (A * 4) أولا ثم بعد ذلك C*(4*A) ثم تجرى عملية الطرح أخيراً منتجة الكية :

```
(B↑2)−(4*A*C).
```

٢ ـــ ٦ استخدام الاقـــواس USE OF PARENTHESES

هناك عدة مواقف تتطلب تغيير التدرج الطبيعي للعمليات الحسابية فى أى صيغة رياضية . ويمكن أن يتم إنجاز ذلك بسهولة بإضافة زوج من الأقواس فى الأماكن الصحيحة بداخل الصيغة الرياضية . وتنفذ العمليات الحسابية بداخل الأقواس الداخلية أولا تتبعها العمليات الحسابية بداخل الأقواس الخارجية ، . . . وهكذا . ولكن سوف يطبق التدرج الطبيعى بداخل زوج من الأقواس إلا إذا تم تعديله بواسطة أزواج أخرى من الأقواس مبيتة داخل الزوج المعطى .

يجب أن نتذكر دائماً استخدام أزواج من الأقواس . إهمال التوازن فى الأقواس اليمنى أو اليسرى من الأخطاء الشائعة .

مثال ٢ – ٦

نفرض أننا نرغب فى حساب قيمة :

$$[2(a+b)^2+(3c)^2]^{m/(n+1)}$$

فإن صيغة البيسك الرياضية التى تناظر هذا الحد الجبرى هى :

```
(2*(A+B)↑2+(3*C)↑2)↑(M/(N+1))
```

إذا كان هناك شك فى ترتيب تنفيذ العمليات الحسابية فيمكننا تقديم زوج إضافى من الأقواس ، وذلك يعطى :

```
((2*((A+B)↑2))+((3*C)↑2))↑(M/(N+1))
```

كلتا الصيغتين الرياضيتين صحيح . ولذلك فيمكن أن نفضل الصيغة الرياضية الأولى ، حيث أنها أقل عدداً فى الأقواس وبالتالى أسهل فى القراءة

٢ ـــ ٧ قواعد خاصة متعلقة بالصيغ الرياضية SPECIAL RULES CONCERNING FORMULAS

يمكن أن تبرز مشاكل خاصة إذا لم تكتب الصيغة الرياضية صحيحة . ويمكن تجنب ذلك إذا طبقنا هذه القواعد :

١ – استهلال المتغير بعلامة (–) مساو لضربه فى 1 –

مثال ٢ – ٧

الصيغة الرياضية :

```
-X↑N
```

مساوية للكية ‎-(X↑N)‎ أو ‎-1*(X↑N)‎ ، وحيث أن الأس له أسبقية على الضرب . ومن ثم إذا كانت قيمة X و N هى 2 و 3 على الترتيب . فإن ‎-X↑N‎ سوف تنتج القيمة 9 – .

٢ – ماعداً الشروط التى تم ذكرها ، فإن العمليات الحسابية لايمكن أن توصف بطريقة ضمنية .

مثال ٢ – ٨

يجب أن يكتب التعبير الجبرى $2(x_1+3x_2)$ فى البيسك (BASIC) كالتالى :

```
2*(X1+3*X2)
```

بتوضيح تام لمعامل الضرب . الصيغتان الرياضيتان ‎(X1 + 3 X2) * 2‎ و ‎2 (X1 + 3 *X2)‎ غير صحيحتين .

٣ – يمكن رفع كية سالبة لقوة إذا كان الأس رقماً صحيحاً (يجب ألا يختلط عليك الأمر بين الأس كصيغة رياضية أسية والأس كجزء من الرقم العشرى) .

ولفهم هذا التقييد ، يجب أن نرى كيف تتم عملية الأس . إذا كان الأس **كية صحيحة** فإن الكية المرفوعة للأس تضرب فى نفسها عدداً مناسباً من المرات .

ومن ناحية أخرى ، نفرض أن الأس **كية عشرية** . فإن الإجراء المتبع مع الأس العشرى هوحساب ، **اللوغاريتم** للكية المطلوبة أن ترفع للأس ثم ضرب هذا اللوغاريتم فى الأس ثم حساب مقابل اللوغاريتم . وحيث أن لوغاريتم الرقم السالب غير معرف ، فإننا نرى أن العملية غير مسموح بها إذا كانت الكية المطلوب رفعها لأس هى كية سالبة .

مثال ٢ — ٩

أدرس الصيغة الرياضية :

```
(C1+C2)↑3
```

تضرب الكية الممثلة (C1 + C2) فى نفسها ثلاث مرات وبذلك ينتج مكعب الكية . لاحظ أنه لايهم إذا كانت الكية (C1 + C2) سالبة أو موجبة

ومع ذلك ، فإن الصيغة الرياضية :

```
(B↑2−4*A*C)↑.5
```

سوف تكون صحيحة ومقبولة فى حالة ما إذا كانت B ↑ 2 − 4 * A * C تمثل بكية موجبة فقط .

وأخيراً ، إدرس ماذا يحدث إذا كان أى من Aأو N فى التعبير A↑N مساوياً لصفر . إذا كانت N تساوى صفراً فإن A↑N سوف تكون قيمتها 1 دون الاكتراث بقيمة A . وإذا كانت قيمة A تساوى صفرا و N قيمة غير الصفر فإن قيمة A↑N تساوى صفراً .

٤ — **لا يمكن اجراء أى عمليات حسابية على سلاسل الحروف أو المتغيرات الحرفية ، ومع ذلك فإن بعض نسخ البيسك تسمح لسلسلة وسلسلة متغيرات أن تتصلا تعاقبياً (أى تتجمع واحدة خلف الأخرى) .**

مثال ٢ — ١٠ :

افترض أن المتغيرين غير الرقميين X$, Y$ قد خصص لهما القيم الآتية :

```
X$="TEN"
Y$="THOUSAND"
```

```
X$+Y$+" DOLLARS"
```

لا تتحقق حيث أنها ليست ذات معنى لتجرى عمليات عددية على سلاسل الحروف . ويكون التعبير فى بعض نسخ البيسك .

```
X$+" "+Y$+" DOLLARS"
```

سوف يتسبب فى ضم السلاسل الحرفيه الثلاث ، وتنتج سلسلة حروف مفرده .

```
TEN THOUSAND DOLLARS
```

٢ — ٨ تحديد قيم — جملة LET   ASSIGNING VALUES—THE LET STATEMENT

تستخدم جملة *LET* لتحديد قيمة رقية أوقيمة حرفية لمتغير . يمكننا تعريف متغير معين فى البرنامج بواسطة تحديد قيمته بهذه الطريقة .

تتكون جملة LET من رقم جملة متبوعة بالكلمة الدالة LET متبوعة بمصطلح تحديد يشبه المعادلة الرياضية . يجب أن يتكون مصطلح التحديد من متغير وعلامة (=) وصيغة رياضية كما هو موضح فى الأمثلة التالية .

مثال ٢ – ١١

```
10 LET X=12.5
20 LET C1=F3
30 LET A=3.141593*R↑2
40 LET N$="NAME"
50 LET T$=N$
```

فى كل من هذه الجمل تعطى قيمة الحد الموجود يمين علامة التساوى للمتغير الموجود على يسار علامة التساوى .

لاحظ أن المتغير الموجود على يسار علامة التساوى والحد الموجود على يمين العلامة يجب أن يكونا من نفس النوع (أما عددية أو حرفية). وبمعنى آخر ، فإن القيمة العددية لايمكن أن نحدد بها قيمة متغير حرفى والعكس صحيح . لاحظ أيضاً أن سلسلة الحروف يجب أن توضع بين علامتى اقتباس إذا ظهرت فى جملة LET .

من الأهمية أن نفهم الفرق بين مصطلح التحديد الذى يظهر فى جملة LET والمعادلة الجبرية . تبدو العديد من مصطلحات التحديد كمعادلات جبرية ومن الناحية الأخرى ، فإن هناك بعض مصطلحات التحديد الجائزة والتى ليس لها أى معنى إذا عوملت معاملة المعادلات الجبرية .

مثال ٢ – ١٢

إدرس جملة LET التالية ، وهى صحيحة وذات معنى :

```
5 LET J=J+1
```

واضح أن مصطلح التحديد $J = J + 1$ لايناظر معادلة جبرية حيث المعادلة $j = j + 1$ ليس لها معنى . ومانعمله هنا هو زيادة قيمة المتغير الرقى J بالوحدة . وبذلك فإن مصطلح التحديد منطقى تماماً إذا فسرناه كالتالى : أضف واحداً للقيمة التى يمثلها المتغير J ثم ضع القيمة الجديدة فى J . لاحظ أن القيمة القديمة للمتغير J سوف تستبدل بالقيمة الجديدة .

سوف ترى أن جمل LET من هذا النوع تستخدم غالباً فى البيسك .

تسمح بعض نسخ البيسك بمرونة أكثر فى كتابة جمل LET عن نسخ أخرى . فمثلا ، فى بعض نسخ البيسك (BASIC) يمكن تحديد نفس القيمة لمتغيرين أو أكثر فى جملة LET واحدة . علاوة على ذلك ، فيمكن حذف الكلمة الدالة LET فى نسخ معينة من البيسك .

مثال ٢ – ١٣

يمكن لجمل LET التالية أن يكون مسموحاً بها فى بعض نسخ البيسك :

```
10 LET A=B=C=5.089
20 A=L*W
30 X1=X2=(A+B)/(C+D)
40 LET A$=K$="TERMINATE"
```

لاحظ أن الجملة الأولى والثالثة والرابعة تتضمن تحديد قيم مضاعفة وأن الكلمة الدالة LET قد تم حذفها من الجملة الثانية والثالثة .

٢ — ٩ قراءة المدخلات — جملة INPUT READING INPUT—THE INPUT STATEMENT

تستخدم جملة *INPUT* لإدخال بيانات رقمية أو حرفية للحاسب خلال تنفيذ البرنامج ، تحتوى الجملة على رقم الجملة ثم الكلمة الدالة INPUT وقائمة من المتغيرات . يمكن أن تتضمن القائمة كلا من المتغيرات الرقمية والحرفية . ويجب فصل المتغيرات عن بعضها بواسطة فصلات (,) .

مثال ٢ — ١٤

```
5 INPUT A,B,C
10 INPUT N$,M$,X0,F5
15 INPUT P(I),Q(I),T$(I)
```

تسمى المتغيرات المبينة فى الجملة الأخيرة **متغيرات ذات أدلة** . سوف نناقش المتغيرات ذات الأدلة فى الفصل الخامس .

عند مقابلة جملة INPUT أثناء تنفيذ البرنامج ، تطبع علامة استفهام (؟) على الكونسول (نضد التشغيل) مشيراً إلى طلب بيانات . وعادة ما تظهر علامة الاستفهام عند أبتداء سطر جديد . يتوقف تنفيذ باقى البرنامج حتى يتم إمداد البرنامج بالبيانات المطلوبة .

يجب أن يعطى المبرمج (أو مستخدم البرنامج) المعلومات المطلوبة فور ظهور علامة الاستفهام بطباعة البيانات الصحيحة على لوحة الكونسول يتبعها رجوع عربة الآلة . بعد ذلك تنقل البيانات إلى ذاكرة الحاسب ويستكمل تنفيذ البرنامج . وبذلك فإن جملة INPUT يمكن أن تكون مفيدة خصوصاً فى أسلوب البرمجة التخاطبى .

يجب ملاحظة القواعد التالية عند إدخال بيانات الإدخال المطلوبة :

١ — يجب أن تناظر بنود البيانات قائمة المتغيرات فى جملة INPUT من حيث العدد والنوع (أى يجب إعطاء الحاسب أرقاماً للمتغيرات الرقمية وحروفاً للمتغيرات الحرفية) وسوف نتجاهل أى بنود بيانات زائدة .

٢ — يجب فصل بنود البيانات عن بعضها بواسطة فصلات (,) .

٣ — يجب أن تتكون بنود البيانات من أرقام وسلاسل حروف وغير مسموح بالصيغ الرياضية .

٤ — يجب حصر سلاسل الحروف المحتوية على فصلات (,) أو المبتدأة بفراغات خالية بين علامات اقتباس " " . يمكن أن تحصر أى سلاسل حرفية أخرى بين علامات اقتباس حسب الرغبة .

مثال ٢ — ١٥

افرض أننا قابلنا هذه الجملة :

```
60 INPUT X,Y,C$
```

أثناء تنفيذ برنامج البيسك ، وسوف يسبب ذلك طباعة علامة استفهام عند بداية سطر جديد على لوحة الكونسول . وسوف يتوقف تنفيذ البرنامج مؤقتا .

وعندما يرى المستفيد علامة الاستفهام ، فسوف يشرع فى إدخال البيانات المطلوبة . نفرض أن القيم الحقيقية للمتغيرات X و Y و C$ هى 5 و -1.2×10^{-3} و " NOVEMBER 27, 1937 ". فيجب أن يظهر سطر بيانات الإدخال كالتالى :

```
? 5,-1.2E-3,"NOVEMBER 27, 1937"
```

بعد طباعة البيانات ، يضغط المستفيد على مفتاح رجوع العربة وذلك يتسبب فى إرسال البيانات إلى ذاكرة الحاسب . ثم يستكمل تنفيذ باقى البرنامج بعد ذلك بالطريقة العادية .

إن جملة INPUT مفيدة للغاية فى البرامج الأولية التى لا تتطلب كيات كبيرة من بيانات الإدخال . ومع ذلك ، فإن إدخال البيانات خلال جملة PRINT فيه استهلاك للوقت نسبياً ، ولا يمكن تخزين البيانات التى تم إدخالها بهذه الطريقة لاستخدام لاحق . (فى معظم أنظمة البيسك يمكن تخزين برنامج لمدة غير محددة وإعادة تشغيله كلما دعت الحاجة إلى ذلك) . سوف نرى طريقة أخرى لتعريف البيانات فى الفصل الخامس .

٢ ــ ١٠ طباعة المخرجات ــ جملة PRINT
PRINTING OUTPUT—THE PRINT STATEMENT

تستخدم جملة *PRINT* لإرسال مخرجات بيانات من الحاسب سواء كانت رقمية أو حرفية وتتكون الجملة من رقم جملة ، ثم الكلمة الدالة PRINT وقائمة من بنود المخرجات . يمكن أن تكون بنود المخرجات أرقاماً أو صيغاً رياضية أو سلاسل حرفية . البنود المتتالية يجب أن تفصل عن بعضها بواسطة إما فصلة (,) أو فصلة منقوطة (;) .

مثال ٢ ــ ١٦

مبين أدناه عدة جمل PRINT نموذجية :

```
100 PRINT A,B,C
110 PRINT "X=";X,"Y=";Y
120 PRINT "NAME:";N$,"ADDRESS:";A$
130 PRINT
140 PRINT K;C$(K);5*X0↑2/2;U(I)+V(I);P$
```

تسمى المتغيرات V(I) و U(I) و C$(K) فى الجملة الأخيرة بمتغيرات ذات أدلة . وسوف نناقش المتغيرات ذات الأدلة فى الفصل الخامس .

يجب اتباع القواعد التالية عند كتابة جملة PRINT .

المباعدة بين الكلمات فى السطر

١ ــ تبدأ كل جملة PRINT سطراً جديداً للمخرجات (تم مناقشة استثناء واحد فى القواعد ٥ و ٦ أدناه) . ومع ذلك ، فسوف ينشأ سطران أو أكثر من المخرجات بواسطة جملة PRINT واحدة إذا كانت قائمة بنود البيانات فيها عدد كبير من المداخل .

مثال ٢ ــ ١٧

سوف تسبب جملة PRINT

```
50 PRINT C1,C2,C3,C4,C5,C6,C7,C8
```

فى طباعة قيم C1، إلى C5 على سطر واحد وطباعة قيم C6 إلى C8 على السطر التالى . ونفرض مثلا أن C1 إلى C8 تمثل بالقيم التالية ·

```
C1=3
C2=-12
C3=6.5
C4=5000
C5=0
C6=.0047
C7=-8
C8=7.2E-15
```

فإن المخرجات سوف تظهر كالتالى :

```
3              -12            6.5            5000
0.0047         -8             7.20000E-15
```

٢ – إذا لم تحتو جملة PRINT على أى بنود بيانات فسوف يظهر سطر خال وهذه طريقة مفيدة للتحكم فى الفراغ الرأسى لبيانات المخرجات .

مثال ٢ – ١٨

سوف تسبب جمل الطباعة التالية :

```
40 PRINT C1,C2,C3,C4
50 PRINT
60 PRINT C5,C6,C7,C8
```

طباعة القيم C1 إلى C4 على سطر واحد وطباعة القيم C5 إلى C8 على سطر آخر بينهما سطر خال .

إذا كانت قيم C1 إلى C8 هى نفس القيم المشار إليها فى مثال ٢ – ١٧ فإن المخرجات الناتجة من جمل PRINT الثلاث السابقة سوف تظهر كالتالى :

```
3              -12            6.5            5000

0              0.0047         -8             7.20000E-15
```

الأرقام المعنوية Significant Figures

٣ – سوف تظهر المخرجات للكميات الرقية كالتالى :

فى معظم نسخ البيسك سوف تطبع أى كية صحيحة تحتوى على 8 خانات أو أقل كرقم صحيح أما إذا تعدى الرقم الصحيح 8 خانات فسوف يقرب إلى 6 أرقام معنوية ويطبع كرقم عشرى له أس .

سوف تطبع الكية العشرية كرقم عشرى . إذا كانت الكمية تحتوى على أكثر من 6 خانات (متضمنا أى أصفار تسبق الرقم يمين العلامة العشرية) فسوف تقرب إلى 6 خانات . سوف تظهر صورة أسية إذا كانت قيمة الرقم تتعدى 999999 أو إذا كانت أقل من 0.1 وتحتوى على أكثر من 6 أرقام معنوية .

مثال ٢ – ١٩

نفرض أن برنامج البيسك يحتوى على المتغيرات A وB وC وD و E وF التى حددت لها القيم التالية :

```
A=1234567
B=123456789
C=-0.001234
D=0.000012345
E=-1234.5
F=1234567.89
```

فسوف تولد الجمل :

```
100 PRINT A,B,C
110 PRINT D,E,F
```

السطران التاليان من المخرجات :

```
1234567        1.23457E+8     -0.001234
1.23450E-5    -1234.5          1.23457E+6
```

سلاسل الحروف Strings

٤ – يجب أن تحصر سلاسل الحروف بين علامتى اقتباس (انظر مثال ٢ – ٢٠ فيما يلى) .

المباعدة بين بنود المخرجات فى السطر Spacing of Output Items Within a Line

٥ – إذا كانت بنود البيانات فى قائمة المخرجات مفصولة عن بعضها بواسطة فصله (,) فإن كل سطر سوف يقسم إلى 5 مناطق متساوية الطول وسوف تطبع قيمة واحدة فى كل منطقة .

مثال ٢ – ٢٠

يحتوى برنامج بيسك على الجملة التالية :

```
65 PRINT "NAME",N$,X,.5*(C1+C2)
```

فإذا كانت المتغيرات قد تم تحديدها بالقيم :

```
N$=CINNAMON        C1=7
X=39               C2=11
```

فإن جملة PRINT السابقة سوف تولد السطر التالى من المخرجات :

```
NAME              CINNAMON          39              9
```

نبين توضيحات أخرى لاستخدام الفصلة (,) فى جملة PRINT . فى الأمثلة ٢ – ١٧ ، ٢ – ١٨ ، ٢ – ١٩ .

إذا تلى البند الأخير من قائمة البيانات فصلة (,) فإن الكمية المخرجة التالية (أى الكمية الأولى من المخرجات فى جملة PRINT اللاحقة) سوف تطبع على نفس السطر فى حالة وجود مسافة كافية . (لاحظ أن هذه تنتج استثناء للقاعدة 1 على الصفحة ٤٠) .

مثال ٢ — ٢١

سوف تسبب الجمل التالية :

```
100 PRINT A,B,C,
110 PRINT D,E,F
```

فى طباعة قيم A وB وC وD وE على سطر واحد وتتبعها قيمة F على السطر التالى .

إذا تم تحديد المتغيرات A وB وC و D وE وF بنفس القيم العددية كما فى المثال ٢ — ١٩ فالمخرجات الناتجة من جمل الطباعة السابقة سوف تظهر كالتالى :

```
1234567         1.23457E+8      -0.001234       1.23450E-5      -1234.5
1.23457E+6
```

(قارن ما سبق بالنتائج فى المثال ٢ — ١٩) .

يمكن أن يصل عدد الفصلات التى تظهر إلى أربعة إذا دعت الحاجة . تأثير كل فصلة هو التحرك إلى بداية منطقة الطباعة التالية . وبذلك يمكن طباعة البيانات بفاصل واسع بينها بهذه الطريقة .

مثال ٢ — ٢٢

برنامج بيسك يحتوى على الجمل التالية :

```
120 PRINT A,B,C,D,E
130 PRINT F,,,,G
```

إذا تم تحديد المتغيرات بالقيم التالية :

```
A=1          C=3          E=5          G=7
B=2          D=4          F=6
```

فإن الجمل السابقة سوف تنتج طباعة السطور التالية :

```
1            2            3            4            5
6                                                   7
```

٦ — إذا تم استخدام الفصلة المنقوطة (;) فضلا عن الفصلة (,) وذلك لفصل بنود البيانات العددية فى قائمة المخرجات فإن قيم المخرجات سوف تظهر أكثر تقارباً من بعضها . سوف تتوقف عدد المسافات المتروكة على عدد أرقام الخانات أو الحروف فى كل بند من بنود المخرجات . وباستخدام الفصلة المنقوطة (;) بهذه الطريقة يمكن طباعة أكثر من 5 كميات على سطر واحد .

مثال ٢ — ٢٣

برنامج بيسك يحتوى على الجملة التالية :

```
100 PRINT A1;A2;A3;A4;A5;A6;A7;A8
```

إذا تم تحديد المتغيرات بالقيم التالية :

```
A1=11        A5=15
A2=12        A6=16
A3=13        A7=17
A4=14        A8=18
```

فإن جملة PRINT السابقة سوف تولد السطر التالى من المخرجات :

```
11   12   13   14   15   16   17   18
```

إذا تبعت سلسلة من الحروف أو متغير حرفى الفصلة المنقوطة (;) فى قائمة المخرجات ، فإن سلسلة الحروف سوف تطبع بدون ترك أى مسافات وسوف يطبع البند التالى من البيانات فوراً بعد سلسلة الحروف .

مثال ٢ – ٢٤

برنامج بيسك يحتوى على الجملة التالية :

```
200 PRINT "X=";X,"Y=";Y
```

إذا تم تحديد المتغيرات بالقيم X = 12 وY = – 5 فإن الجملة السابقة سوف تولد سطر المخرجات التالى :

```
X=  12          Y=-5
```

تسبق جملة PRINT فى معظم برامج البيسك جملة INPUT وتحتوى على سلسلة حروف . والغرض من جملة INPUT هو إصدار رسالة سريعة لطلب البيانات المطلوبة . إذا تبعت سلسلة الحروف فصلة منقوطة (;) فإن علامة الاستفهام الناتجة من جملة INPUT سوف تظهر عند نهاية الرسالة المطبوعة .

مثال ٢ – ٢٥

تم كتابة برنامج بيسك لحساب مساحة ومحيط دائرة . أول خطوة عند تنفيذ البرنامج هى قراءة قيمة نصف القطر . حيث يحتوى البرنامج على الجمل التالية :

```
10 PRINT "RADIUS=";
20 INPUT R
```

وسوف تسبب هاتان الجملتان طباعة السطر التالى من المخرجات .

```
RADIUS=?
```

وبعد ذلك يدخل المستفيد قيمة R (نصف القطر) كما هو موضح فى المثال ١ – ٥ .

وأخيراً ، يجب أن يفهم أن تأثير وضع الفصلة المنقوطة (;) بعد آخر مدخل من قائمة البيانات هو تماماً نفس التأثير عند وضع الفصلة (,) فى هذا المكان (أى أن الكمية التالية فى الطباعة سوف تظهر على نفس السطر) وقد تم توضيح ذلك فى مثال ٢ – ٢٥ .

٢ – ١١ جملة END   THE END STATEMENT

تشير جملة END إلى نهاية برنامج بيسك . تحتوى الجملة ببساطة على رقم الجملة ، تتبعها الكلمة الدالة END . وهذه الجملة مطلوبة فى نسخ بيسك القديمة (وهى اختيارية فى بعض النسخ الحديثة) ، وإذا ما طلبت فلابد أن تكون آخر جملة فى البرنامج ، كما يجب أن تحمل أعلى رقم جملة فى البرنامج .

أحد استخدامات جملة END موضح فى مثال ١ – ٦ . انظر مثال ٢ – ٢٦ لترى توضيحاً آخر لذلك .

٢ ــ ١٢ كتابة برامج كاملة بلغة البيسك WRITING COMPLETE BASIC PROGRAMS

تعلمنا حتى الآن كيفية قراءة بيانات للحاسب وتنفيذ حسابات رياضية وكتابة النتائج . وبذلك يمكننا القيام بكل الخطوات الهامة فى برنامج بيسك كامل (ولكن بسيط) .

فى الفصل الثالث ، سوف نناقش ميكانيكية إدخال برنامج للحاسب ، وتنقيح البرنامج ثم تنفيذه ، ولكن سوف نهتم الآن بكتابة برامج بسيطة ويوضح مثال ٢ – ٢٦ مثل هذه البرامج . ونحن نحث القارئ لكتابة برامج قليلة أخرى خاصة به وتحمل نفس الطبيعة . (اقتر حات عديدة معطاة فى نهاية هذا الفصل) .

مثال ٢ – ٢٦ جذور المعادلة التربيعية

نرغب فى حساب جذور المعادلة التربيعية باستخدام الصيغة الرياضية المعروفة .

$$x_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a} \qquad x_2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

ولنفرض أن قيم a و b و c كلها تجعل $b^2 - 4ac$ دائماً قيمة موجبة وبذلك لا نبالى عند حساب الجذر التربيعى بقيمة سالبة .

الخطوات التى يجب أن تتبع هى :

١ – قراءة القيم الرقمية a و b و c

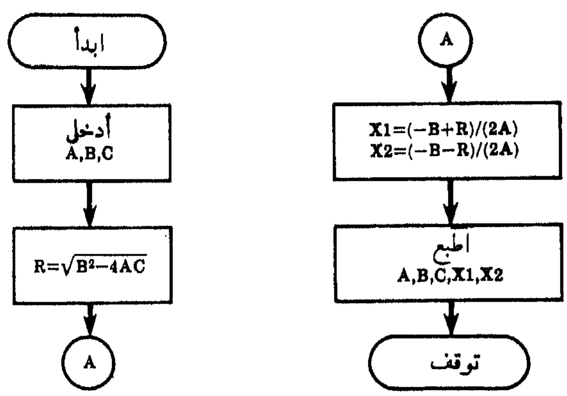
٢ – حساب قيمة $\sqrt{b^2 - 4ac}$

٣ – حساب قيم x_1 و x_2 باستخدام الصيغ الرياضية السابقة .

٤ – طباعة قيم a و b و c و x_1 و x_2

٥ – توقف .

ويبين شكل ٢ – ١ خريطة سير العمليات المناظرة لذلك



شكل ٢ – ١

إذا استخدمنا المواد التي تم تقديمها سابقاً فى هذا الفصل ، فكتابة برنامج كامل يقوم بهذه الحسابات يكون سهلا للغاية . ويمكن تنفيذ الخطوة الأولى بواسطة جملة PRINT و جملة INPUT والخطوتان ٢ ، ٣ باستخدام جمل LET. وجمل PRINT تصبح مطلوبة لتنفيذ الخطوة ٤ وجملة END لخطوة ٥ . يبين شكل ٢ – ٢ برنامج البيسك المطلوب، لاحظ أن بيانات الادخال تحتها خط .

```
10 PRINT "ENTER VALUES FOR A, B AND C"
20 INPUT A,B,C
30 LET R=(B^2-4*A*C)^.5
40 LET X1=(-B+R)/(2*A)
50 LET X2=(-B-R)/(2*A)
60 PRINT
70 PRINT "A=";A,"B=";B,"C=";C
80 PRINT "X1=";X1,"X2=";X2
90 END

>RUN

ENTER VALUES FOR A, B AND C
 ?2,5,3

A= 2           B= 5          C= 3
X1=-1          X2=-1.5
```

شكل ٢ – ٢

فى نهاية الشكل ٢ – ٢ تظهر المخرجات المتولدة بواسطة البرنامج $a = 2$ و $b = 5$ و $c = 3$ (سوف تناقش طريقة تنفيذ برنامج بيسك فى الفصل الثالث) نرى أن قيمة x_1 هى -1 وقيمة x_2 هى -1.5

٢ – ١٣ التعليقات على البرنامج – جملة REM

PROGRAM COMMENTS—THE REM STATEMENT

الطريقة الأكثر شيوعاً لتقديم ملاحظات (تعليقات) فى برنامج بيسك هى استخدام جملة (*REMARK*) . تحتوى هذه الجملة على رقم جملة تتبعها الكلمة الدالة REM ورسالة نصية . يمكن إضافة جمل REM فى أى مكان من برنامج بيسك .

مثال ٢ – ٢٧

ونبين فيما يلي جملة REM نموذجية :

```
5 REM PROGRAM TO CALCULATE THE ROOTS OF A QUADRATIC EQUATION
```

سوف تمدنا هذه الجملة بعنوان مناسب للبرنامج المعروض فى مثال ٢ – ٢٦ .

لا تعطى كلمة REM الحاسب أى أمر للتنفيذ . ولكنها مع ذلك ستظهر فى القائمة مع كل الجمل الأخرى الموجودة فى برنامج بيسك بالتسلسل الصحيح . وبذلك فإنها تعرض على المبرمج طريقة مناسبة لتوثيق البرنامج (أى تعطى عنواناً للبرنامج لتعريف المتغيرات المهمة وتميز الأجزاء المنطقية الرئيسية من البرنامج) . وسوف نرى توضيحات كثيرة لاستخدام جمل REM فى أمثلة لاحقة .

يترتب على إحاطة جملة REM بسطور خالية ظهور جملة REM بوضوح عن باقى البرنامج ، وذلك يضيف الوضوح إلى توثيق البرنامج .

فى بعض الأحيان يتطلب إضافة تعليق يوضح جملة لها دلالة معينة فى برنامج بيسك ويمكن إنجاز ذلك بالطبع باستخدام جملة REM ، ومع ذلك يمكن أن يكون وضع التعليق على نفس السطر الموجود بها الجملة طريقة أكثر تفضيلا . يجب أن يكون التعليق من هذا النوع مسبوقاً بفصلة عليها (،) وذلك لتمييزها عن نهاية الجملة . مثل هذا التعليق لا يمكن أن يتعدى الطول الباقى من السطر .

مثال ٢ — ٢٨

أضف التعليقين CALCULATE SECOND ROOT, CALCULATE FIRST ROOT للجملة الرابعة والخامسة من البرنامج المبين فى شكل ٢ — ٢ على الترتيب .

سوف تظهر الجملتان الرابعة والخامسة كالتالى :

```
40 LET X1=(-B+R)/(2*A)  'CALCULATE FIRST ROOT
50 LET X2=(-B-R)/(2*A)  'CALCULATE SECOND ROOT
```

٢ — ١٤ تحويل التحكم — جملة GO TO

TRANSFERRING CONTROL—THE GO TO STATEMENT

عادة ما يتم تنفيذ جمل برنامج بيسك بنفس ترتيب ظهورها واحدة تلو الأخرى . ومع ذلك ، فى بعض الأحيان يصبح من الضرورى أن « نقفز » إلى بعض الأجزاء الأخرى من البرنامج . وبذلك نغير التسلسل الطبيعى للتنفيذ . ويمكن إنجاز ذلك بواسطة **جملة *GO TO*** . ودائماً ما نشير إلى مثل هذا القفز كعملية **تفرع غير مشروط** أو **تحويل التحكم** . وعلى ذلك فجملة GO TO تسمح لنا بتحويل التحكم لأى جملة أخرى فى برنامج بيسك (بما فى ذلك جملة REM) .

تحتوى جملة GO TO على رقم جملة تتبعها الكلمة الدالة GO TO ورقم السطر أو الجملة التى ينتقل إليها التحكم .

مثال ٢ — ٢٩

يحتوى برنامج بيسك على هذه الجملة التالية :

```
100 GO TO 10
```

وبذلك فإن الحاسب يؤمر بتنفيذ الجملة رقم 10 بعد ذلك .

وسوف نهتم بعمليات التفرع بطريقة تفصيلية فى الفصل الرابع . أما الآن فسوف نقصر اهتمامنا لتطبيق بسيط وهام لجملة GO TO .

٢ — ١٥ تكرار تنفيذ البرنامج REPETITIOUS PROGRAM EXECUTION

تنجم مواقف كثيرة تتطلب استخدام برنامج بيسك لتشغيل عدة مجموعات من البيانات واحدة تلو الأخرى . ويمكن إنجاز ذلك ببساطة بإنهاء البرنامج بتحويل التحكم مرة ثانية لجملة « قراءة المدخلات » ، وذلك يسبب قراءة مجموعة متتابعة من بيانات الإدخال للحاسب ثم تشغيلها . (لاحظ أننا نشير إلى النهاية **المنطقية** للبرنامج وليس النهاية **المعنوية** . آخر جملة **معنوية** فى البرنامج يجب أن تكون أيضاً جملة END) . سوف نستمر فى هذا الإجراء حتى يتم الانتهاء من تشغيل كل بيانات الإدخال ، عند هذا الوقت يجب على المستفيد بها أن ينهى اتصاله بالحاسب .

دائماً يتم تحويل التحكم بواسطة جملة GO TO وهذا موضح فى مثال ٢ — ٣٠ .

مثال ٢ — ٣٠

دعنا نعدل فى برنامج البيسك المعروض فى المثال ٢ — ٢٦ وبذلك يمكننا تشغيل عدة مجموعات من بيانات الإدخال (أى أنصاف أقطار مختلفة) على التوالى . إذا فحصنا البرنامج المبين فى شكل ٢ — ٢ سنرى أنه من الأسهل أن نقوم بهذه التعديلات إذا أضفنا

جملة "GO TO 10" قبل جملة END مباشرة .

يبين شكل ٢ — ٣ البرنامج المعدل . لاحظ أننا أضفنا جملة PRINT خالية وجملة GO TO 10 قرب نهاية البرنامج . والغرض من جملة PRINT الخالية هي فصل بيانات الإخراج عن رسائل المدخلات اللاحقة . ولاحظ أيضا ، أننا قد أضفنا جملة REM عند بداية البرنامج وقد تم إضافة هذه التعليقات فى السطور 40 و 50 لاحظ أن المعلومات التى أدخلها المستخدم قد وضع تحتها خط .

```
5   REM PROGRAM TO CALCULATE THE ROOTS OF A QUADRATIC EQUATION
10 PRINT "ENTER VALUES FOR A, B AND C"
20 INPUT A,B,C
30 LET R=(B^2-4*A*C)^.5
40 LET X1=(-B+R)/(2*A)     'CALCULATE FIRST ROOT
50 LET X2=(-B-R)/(2*A)     'CALCULATE SECOND ROOT
60 PRINT
70 PRINT "A=";A,"B=";B,"C=";C
80 PRINT "X1=";X1,"X2=";X2
90 PRINT
100 GOTO 10
110 END

>RUN

ENTER VALUES FOR A, B AND C
 ?2,6,1

A= 2           B= 6           C= 1
X1=-0.177124   X2=-2.82288

ENTER VALUES FOR A, B AND C
 ?3,3,0

A= 3           B= 3           C= 0
X1=-9.93411E-9                X2=-1.

ENTER VALUES FOR A, B AND C
 ?1,3,1

A= 1           B= 3           C= 1
X1=-0.381966   X2=-2.61803
```

شكل ٢ — ٣

نرى المخرجات الناتجة من ثلاث مجموعات مختلفة لقيم a وb وc فى نهاية شكل ٢ — ٣ . (لاحظ أن القيمة التالية لـ x_1 محسوبة -9.9×10^{-9} وليست صفراً ، وهو الحل الصحيح) . ينتهى الاتصال بالحاسب بعد تشغيل ثالث مجموعة من البيانات ، بالرغم من إمكانية استمرار الإجراءات طالما رغبنا فى ذلك .

٢ — ١٦ ملاحظـــــات ختامية CLOSING REMARKS

تعلمنا حتى الآن القدر الكافى عن البيسك والذى يمكن المستفيد من تنظيم وكتابة برامج كاملة خاصة به بالرغم من بساطتها . والفصول القادمة سوف تبين كيفية كتابة برامج أكثر تشويقاً فيها شىء من التحدى والتعقيد .

اسئلة للمراجعة

Review Questions

٢ – ١ اذكر طريقتين لكتابة الأرقام (الثوابت) فى البيسك .

٢ – ٢ لخص القواعد النحوية لكتابة الأرقام .

٢ – ٣ اعرض مقارنة تفصيلية بين رقم مكتوب بالترميز العلمى ورقم آخر مكتوب لكمية عشرية مرفوعة لأس فى لغة البيسك .

٢ – ٤ ما هى سلسلة الحروف ؟ ولماذا تستخدم سلاسل الحروف هذه ؟

٢ – ٥ لخص القواعد النحوية لكتابة المتغيرات الرقية والحرفية .

٢ – ٦ ما هى المعاملات الرياضية المستخدمة فى البيسك ؟ وما هو التدرج الهرمى الطبيعى لها ؟ وما هو الترتيب الذى تتم به العمليات الحسابية بداخل مجموعة متدرجة ؟

٢ – ٧ ما هى الصيغة الرياضية (التعبير الرياضى) فى لغة البيسك وماذا تمثل الصيغة الرياضية ؟

٢ – ٨ كيف يمكن تغيير التدرج الهرمى الطبيعى للعمليات الحسابية فى صيغة رياضية ؟

٢ – ٩ اذكر مشكلة معينة يمكن أن تظهر فى عمليات الأس الحسابية . اعرض سبب المشكلة ثم صف كيف يمكن أن نتجنب مثل هذه المشكلة .

٢ – ١٠ ما هو الغرض من جملة LET ؟

٢ – ١١ لخص القواعد النحوية لكتابة جملة LET ؟

٢ – ١٢ ناقش أوجه الشبه والاختلافات بين جملة LET والمعادلة الجبرية ؟

٢ – ١٣ بأى شكل تكون قواعد كتابة جملة LET أقل صرامة نجو بعض نسخ البيسك ؟

٢ – ١٤ ما هو الغرض من جملة INPUT ؟

٢ – ١٥ ماذا يحدث عند مقابلة جملة INPUT أثناء تنفيذ برنامج بيسك ؟

٢ – ١٦ لخص القواعد النحوية لكتابة جملة INPUT .

٢ – ١٧ اذكر ميزتين لاستخدام جمل INPUT لإدخال البيانات للحاسب .

٢ – ١٨ ما هو الغرض من جملة PRINT ؟

٢ – ١٩ لخص القواعد النحوية التى تطبق على جمل الطباعة التالية :

(ا) توليد ومباعدة المسافات بين سطور المخرجات .
(ب) كيفية ظهور الكميات العددية والعدد الأقصى للأرقام المنوية .
(جـ) معاملة السلاسل الحرفية .
(د) المباعدة بين الكميات العددية وسلاسل الحروف بداخل سطر من المخرجات .

٢ – ٢٠ بأى طريقة يمكن استخدام جملة PRINT مقترنة بجملة INPUT عند قراءة بيانات للحاسب ؟

٢ – ٢١ ما هو الغرض من جملة END ؟ ما هى القواعد التى تزامل استخدامها ؟

٢ – ٢٢ ما هو الغرض من جملة REM ؟ ما هى القواعد التى تحكم استخدامها ؟

٢ – ٢٣ ما هو المقصود من توثيق البرنامج ؟ كيف يمكن القيام بتوثيق برنامج فى البيسك ؟

٢ – ٢٤ ما هو الغرض من جملة GO TO ؟ وكيف تكتب ؟

٢ – ٢٥ اذكر اسم جملة واحدة يجب أن توجد فى كل برنامج من برامج البيسك عند استخدام إحدى نسخ بيسك القديمة وأين تظهر هذه الجملة ؟ وماذا يمكن أن يقال عن رقم الجملة الخاصة بها ؟

٢ – ٢٦ ما هى الميزة التى توجد بالتحديد عند كتابة برنامج بيسك يمكن تنفيذه عدة مرات ؟ وهل يتطلب كية هائلة من الجهد فى البرمجة عند كتابة برنامج بهذه الطريقة ؟

مســائل محلولة

Solved Problems

٢ – ٢٧ عبر عن كل من الكميات التالية كأرقام يمكن تمثيلها بلغة البيسك .

الكمية	الرقم بالبيسك
7,350	7350 or 7.35E+3
−12	−12
10^6	1000000 or 1E+6
$-2{,}053.18 \times 10^3$	−2053180 or −2.05318E+6
0.00008291	0.00008291 or 8.291E−5
9.563×10^{12}	9.563E+12
1/6	0.16666667

٢ – ٢٨ كتبت أرقام البيسك التالية بطريقة غير صحيحة . تعرف على الأخطاء .

الرقم	الخطأ
7,104	غير مسموح بالفصلة (،)
−+4920	غير مسموح بعلامتين متتاليتين (+ ، -)
2.665E+42	قيمة الأس كبيرة جداً
0.333333333333	أرقام معنوية كثيرة جداً
4.63E−0.8	لا يمكن للأس أن يحتوى على علامة عشرية

٢ – ٢٩ تمثل كل من البنود التالية سلسلة من الحروف ممثلة فى لغة بيسك تعرف إذا كان أى منها مكتوباً بطريقة غير صحيحة .

سلسلة حروف	الخطأ
TWENTY-SEVEN	صحيحة
2+5=7	صحيحة
ENTER ALL INPUT DATA	طويلة فى بعض نسخ بيسك
75.50	صحيحة
SYMBOL IS "X"	غير مسموح بعلامات الاقتباس

٢ – ٣٠ يمثل كل من البنود التالية متغيراً عددياً . تعرف إذا كان أى منها مكتوباً بطريقة غير صحيحة .

المتغير	الخطأ
XR	يجب أن يكون الحرف الثانى إن وجد رقماً صحيحاً .
Q	صحيحة
C23	حروف كثيرة
8C	الحرف الأول يجب أن يكون حرفاً هجائياً وإن وجد حرف آخر يجب أن يكون رقما صحيحاً .
BIGC	حروف كثيرة
J8	صحيحة
A$	يجب أن يكون الحرف الثانى إن وجد رقماً صحيحاً .

٢ – ٣١ يمثل كل من البنود التالية متغيراً حرفياً . تعرف إذا كان أى منها مكتوباً بطريقة غير صحيحة .

المتغير	الخطأ
N$	صحيحة
C	علامة الدولار غير موجودة .
Z$3	آخر حرف يجب أن يكون علامة الدولار وبعض نسخ بيسك تسمح بحرفين فقط .
Z3$	يمكن أن يكون صحيحاً بالرغم من أن بعض نسخ بيسك تسمح بحرفين فقط .
E$	صحيحة .

٢ – ٣٢ اكتب صيغة رياضية بلغة بيسك تناظر كلا من التعبيرات الجبرية التالية .

التعبيرات الجبرية	صيغة بيسك
$3x + 5$	3*X+5
$i + j - 2$	I+J−2
$x^2 + y^2$	X↑2+Y↑2
$(x + y)^2$	(X+Y)↑2
$a/b + c/d$	A/B+C/D or (A/B)+(C/D)
$(u + v)^{k-1}$	(U+V)↑(K−1)
$(4t)^{1/6}$	(4*T)↑.16666667 or (4*T)↑(1/6)

٢ – ٣٣ اكتب جملة LET لكل من المواقف التالية :

(ا) حدد قيمة المتغير c بالرقم 2.54 .

```
10 LET C=2.54
```

(ب) حدد قيمة المتغير X بالرقم 12 .

```
20 LET X=12
```

(ج) حدد قيمة المتغير N1 بالقيمة الممثلة بالمتغير N .

```
30 LET N1=N
```

(د) حدد قيمة المتغير A $ بالسلسلة الحرفية JANUARY 31 .

```
40 LET A$="JANUARY 31"
```

(هـ) حدد قيمة المتغير T$ بالسلسلة الحرفية الممثلة بالمتغير S$.

```
50 LET T$=S$
```

(و) حدد قيمة المتغير F بالقيمة الممثلة بالصيغة الرياضية (A↑2+B↑2+C↑2).

```
60 LET F=A↑2+B↑2+C↑2
```

(ز) أضف 0.01 على القيمة الموجودة فى المتغير C7 .

```
70 LET C7=C7+.01
```

(ح) حدد قيمة المتغير I بالقيمة الممثلة بالصيغة الرياضية (I + J) .

```
80 LET I=I+J
```

٢ – ٣٤ اكتب جملة LET المتعددة لكل من المواقف التالية :

(ا) حدد قيمة المتغيرات C1 وC2 وC3 بالقيمة 37.5−

```
10 LET C1=C2=C3=−37.5
```

(ب) حدد قيمة المتغيرين P5$ و P7$ بالسلسلة الحرفية : ***** ERROR *****

```
20 LET P5$=P7$="*****ERROR*****"
```

(جـ) حدد قيمة المتغيرين M و N بالقيمة الممثلة بالصيغة الرياضية : (I + J)/K

```
30 LET M=N=(I+J)/K
```

لاحظ أن جملة LET المتعددة غير متاحة فى كل نسخ بيسك وكذلك يمكن حذف الكلمة الدالة LET فى بعض نسخ بيسك ، أى

```
30 M=N=(I+J)/K
```

٢ – ٣٥ اكتب جملة LET تناظر كلا من المعادلات الجبرية التالية :

(ا) $z = (x/y) + 3$

```
10 LET Z=X/Y+3
```

(ب) $z = x/(y + 3)$

```
20 LET Z=X/(Y+3)
```

(جـ) $w = (u + v)/(s + t)$

```
30 LET W=(U+V)/(S+T)
```

(د) $f = \left[\dfrac{2ab}{c+1} - \dfrac{t}{3(p+q)}\right]^{1/3}$

```
40 LET F=(2*A*B/(C+1)−T/(3*(P+Q)))↑.33333333
```

(هـ) $r = \dfrac{6.8(a+b)^2/c - 7.2a/\sqrt{b+c}}{(a+c)^{1/n}}$

```
50 LET R=(6.8*(A+B)↑2/C−7.2*A/(B+C)↑.5)/(A+C)↑(1/N)
```

٢ – ٣٦ فيما يلي معادلتان جبريتان معقدتان . استبدل كل معادلة بعدة معادلات بسيطة واكتب جملة LET المناظرة لها .

(أ) $$t = \left[\frac{2ab}{c+1} - \frac{r}{7(p+q)}\right]^{1/n}$$

$t_1 = \dfrac{2ab}{c+1}$ — `10 LET T1=2*A*B/(C+1)`

$t_2 = \dfrac{r}{7(p+q)}$ — `20 LET T2=R/(7*(P+Q))`

$t = (t_1 - t_2)^{1/n}$ — `30 LET T=(T1-T2)↑(1/N)`

(ب) $$f = \frac{[6.8(a-b)^2/c - 7.2a/\sqrt{b+c}]^{1/7}}{[(c-a)^m + b^n]^{1/3}}$$

$f_1 = 6.8(a-b)^2/c$ — `10 LET F1=6.8*(A-B)↑2/C`

$f_2 = 7.2a/\sqrt{b+c}$ — `20 LET F2=7.2*A/(B+C)↑.5`

$f_3 = (c-a)^m + b^n$ — `30 LET F3=(C-A)↑M+B↑N`

$f = (f_1 - f_2)^{1/7}/f_3^{1/3}$ — `40 LET F=(F1-F2)↑(1/7)/F3↑(1/3)` or

```
40 LET F=(F1-F2)↑.14285714/F3↑.33333333
```

٢ – ٣٧ اكتب جملة مناسبة أو عدة جمل لكل من المواقف الموصوفة فيما يلي :

(أ) أدخل قيما عددية لكل من X1 وX2 وX3 وقيمة حرفية للمتغير X$ يجب أن تطبع كل البيانات على سطر واحد .

```
10 INPUT X1,X2,X3,X$
```

(ب) أدخل قيما عددية لكل من X1 وX2 وX3 على سطر واحد وقيمة حرفية للمتغير X$ على السطر التالى .

```
10 INPUT X1,X2,X3
15 INPUT X$
```

(جـ) أدخل قيما عددية لكل من X1 وX2 على سطر واحد وقيمة عددية للمتغير X3 تتبعها قيمة حرفية للمتغير X$ على السطر التالى .

```
10 INPUT X1,X2
15 INPUT X3,X$
```

(د) اطبع قيم المتغيرات C1 وC2 وC3 وC4 وC5 كلها على سطر واحد .

```
50 PRINT C1,C2,C3,C4,C5
```

(هـ) اطبع قيم المتغيرات A1 وA2 وA3 على سطر واحد وقيم المتغيرات B1 وB2 وB3 على سطر آخر ويفصل بينهما سطر خال .

```
60 PRINT A1,A2,A3
65 PRINT
70 PRINT B1,B2,B3
```

(و) اطبع قيم المتغيرات A1 وA2 و A3 وB1 وB2 وB3 كلها فى سطر واحد متقاربة من بعضها البعض بقدر المستطاع .

```
50 PRINT A1;A2;A3;A4;A5;A6
```

(ز) اطبع قيم المتغيرات X و Y وZ على سطر واحد ، اسبق كل قيمة عددية بعنوان مناسب .

```
100 PRINT "X=";X,"Y=";Y,"Z=";Z
```

أو

```
100 PRINT "X=";X;"Y=";Y;"Z=";Z
```

(ح) اطبع قيمة المتغير N$ تتلوها قيمة N ثم بعد ذلك قيمة الصيغة الرياضية و A↑2+B↑2

```
120 PRINT N$;N;A↑2+B↑2
```

or أو

```
120 PRINT N$;N,A↑2+B↑2
```

(ط) اطبع السلسلتين الحرفيتين LEFT وRIGHT قرب الحافة اليمنى والحافة اليسرى .

```
150 PRINT "LEFT",,,,"RIGHT"
```

(ى) اطبع الرسالة ROOTS OF SIMULTANEOUS EQUATIONS على سطر واحد فى الوسط بقدر المستطاع .

```
200 PRINT, "ROOTS OF SIMULTANEOUS EQUATIONS"
```

(ك) اطلب رسالة تشير إلى طلب قيمة عددية للمتغير C ، ثم بعد ذلك أدخل القيمة العددية للمتغير C .

```
5 PRINT "C=";
10 INPUT C
```

٢ – ٣٨ بين كيف تظهر البيانات المدخلة فى كل من المواقف التالية :

(ا)

```
10 INPUT X1,X2,X3,X$
```

حيث $X1=4.83\times10^{-3}$ X3=941.55

X2=−537 X$=BUCS

```
?4.83E−3,−537,941.55,BUCS
```

أو

```
?.00483,−537,941.55,BUCS
```

(ب)

```
10 INPUT X$,X1
20 INPUT X2,X3
```

حيث تكون المتغيرات لها نفس القيم الموجودة فى الجزء (ا)

```
?BUCS,4.83E−3
?−537,941.55
```

(ج)

```
30 INPUT A,A$,A1
```

حيث A=350 A1=−8.05

A$=APRIL 12, 1969

```
?350,"APRIL 12, 1969",−8.05
```

٢ – ٣٩ بين كيف تظهر المخرجات المطبوعة فى كل من المواقف التالية

(ا)

```
10 PRINT "NAME ",N$,(X+Y)↑2/3,T4
```

حيث N$=GEORGE Y=8.2

X=27.6 $T4=-5.83\times10^{-4}$

```
NAME          GEORGE          427.213          -0.000583
```

(ب)

```
10 PRINT "NAME ";N$;(X+Y)↑2/3;T4
```

حيث تكون المتغيرات لها نفس القيم الموجودة فى الجزء (أ)

```
NAME GEØRGE 427.213 -0.000583
```

(جـ)

```
12 PRINT A1,A2,A3,A4
14 PRINT B1,B2,B3,B4
```

حيث

$A1=7.43 \times 10^3$	$B1=-2.55 \times 10^{-8}$
$A2=-4373665.8$	$B2=0.843 \times 10^7$
$A3=0.0006066183$	$B3=400.33$
$A4=-3136687$	$B4=10^{-3}$

```
 7430            -4.37367E+6     6.06618E-4     -3136687
-2.55000E-8       8430000        400.33          0.001
```

(د)

```
12 PRINT A1;A2;A3;A4;
14 PRINT B1;B2;B3;B4
```

حيث تكون المتغيرات لها نفس القيم الموجودة فى الجزء (جـ)

```
7430 -4.37367E+6  6.06618E-4 -3136687 -2.55000E-8  8430000  400.33
0.001
```

٢ – ٤٠ بين كيف يمكن وضع التعليق (الملاحظات) فى برنامج بيسك فى كل من المجالات التالية :

(أ) أضف عنوان البرنامج AREA AND CIRCUMFERENCE OF A CIRCLE

```
10 REM AREA AND CIRCUMFERENCE OF A CIRCLE
```

(ب) أضف التعليقات AREA و CIRCUMFERENCE للجمل التالية :

```
40 LET A=P*R↑2
```

و

```
50 LET C=2*P*R
```

```
40 LET A=P*R↑2   'AREA
50 LET C=2*P*R   'CIRCUMFERENCE
```

٢ – ٤١ عدة جمل GO TO مبينة فيما يلى . تعرف إذا كان أى منها مكتوباً بطريقة غير صحيحة .

الجملة	الخطأ
10 GO TO 50	صحيحة
120 GO TO M	رقم الجملة التى يتحول إليها التحكم يجب أن يكون رقماً صحيحاً موجباً وليس متغيرا .
80 GO TO 25	صحيحة
50 GO TO 50	لا يمكن لجملة GO TO أن تحول التحكم إلى نفسها .

مسائل تكميلية
Supplementary Problems

٢ – ٤٢ أجب عن الأسئلة التالية بما هو متاح فى نسخ بيسك المستخدمة فى مدرستك أو مكتبك .

(ا) كم عدد الأرقام المعنوية التى يمكن أن يتضمنها أى عدد ؟

(ب) كم عدد الحروف التى يمكن أن تظهر فى سلسلة حرفية (أى ما هو أقصى طول للسلسلة الحرفية) ؟

(جـ) هل يمكن أن يكتب متغير حرفى كحرف يتبعه رقم ثم يتبعه علامة الدولار ؟ (مثلا C1$)

(د) هل يمكن حذف الكلمة الدالة LET من جملة LET (مثلا 10 A = B + C)

(هـ) هل مسموح بتحديد قيم لأكثر من متغير فى جملة LET واحدة (مثلا 10 LET X = Y = Z = 13) ؟

٢ – ٤٣ عبر عن كل من الكميات التالية كعدد من أعداد بيسك .

(ا)	5	(هـ)	−7,328,500
(ب)	8000	(و)	0.2851×10^4
(جـ)	-1.8033×10^{-9}	(ز)	0.2851×10^{10}
(د)	1/3	(ح)	−16,752.47

٢ – ٤٤ بعض الثوابت التالية كتبت بطريقة خاطئة . تعرف على كل الأخطاء .

(ا)	+0.250	(هـ)	−77777777	(ط)	0.833333333333E−2
(ب)	5076	(و)	1,000,000	(ى)	−00263
(جـ)	3 E−2	(ز)	2.53E+99	(ك)	4.48E
(د)	3.8822E−7.3	(ح)	64E+6	(ل)	0.83333333−E2

٢ – ٤٥ بعض السلاسل الحرفية التالية كتبت بطريقة خاطئة . تعرف على كل الأخطاء .

(ا) $1,995.00

(ب) JULY 4, 1776

(جـ) BEGINNER'S ALL-PURPOSE SYMBOLIC INSTRUCTION CODE

(د) 4 O'CLOCK

(هـ) "NUTS!"

(و) 2X+4Y=Z

٢ – ٤٦ يمثل كل متغير من المتغيرات التالية إما متغير رقمى أو متغير حرفى ولكن بعضها مكتوب بطريقة خاطئة . حدد نوع المتغير فى كل حالة صحيحة ثم تعرف على كل الأخطاء .

(ا)	J	(جـ)	J$	(هـ)	$J6	(ز)	XSTAR	(ط)	PI	(ك)	5T	(م)	Z0
(ب)	J6	(د)	J$6	(و)	J6$	(ح)	C10	(ى)	N$	(ل)	Y*	(ن)	M 5

٢ – ٤٧ اكتب صيغة رياضية تناظر كلا من الحدود الجبرية التالية :

(ا) t^{n+1}

(ب) $(x+3)^{1/k}$

(جـ) $2(a/b)^{1/3}$

(د) $1.87(u+v) - 5.088(x/y + 2z^2)$

(هـ) $1 - x + x^2/2 - x^3/6 + x^4/24 - x^5/120$

(و) $\dfrac{2(p/q)^{k-1}}{(r-3t)^{1/m}}$

(ز) $(i+j-1)^2/5$

(ح) $\left[\dfrac{(x_1+x_2)^m(y_1+y_2)^n}{(x_1/y_1)^{m+n}(x_2/y_2)^{m-n}}\right]^{1/mn}$

٢ – ٤٨ فى نسخة البيسك التى لديك ، هل يمكن لمتغير لم يتم تعريفه (أى لم تخصص له قيمة) أن يظهر على الطرف الأيمن من علامة التساوى فى جملة LET ؟ وإذا ظهر فما هو تأثير ذلك ؟

٢ – ٤٩ اكتب جملة LET لكل من المواقف التالية :

(ا) حدد قيمة المتغير P بالرقم 758.33

(ب) حدد قيمة المتغير B بالقيمة الممثلة بالمتغير A .

(جـ) حدد قيمة المتغير F$ بالقيمة PITTSBURGH , PA

(د) حدد قيمة المتغير N$ بالقيمة الممثلة فى المتغير M$

(هـ) حدد قيمة المتغير Y3 بالصيغة الرياضية X / (A + B + C)

(و) انقص 2 من القيمة الموجودة فى المتغير K .

(ز) ضاعف القيمة الموجودة فى المتغير C5 .

(ح) حدد قيمتى المتغيرين B و C بقيمة الصيغة الرياضية : (A↑2+B↑2)↑.5

٢ – ٥٠ اكتب جملة LET التى تناظر كلا من المعادلات الجبرية التالية :

(ا) $$w = \frac{(a+3)b^n}{2.7(c-d/b)+1}$$

(ب) $$f = \left\{\frac{(a/b)^n/(c-d)^m}{[d/(b-a)^{n+m}]}\right\}^{1/(n+m)}$$

(جـ) $$y = \frac{a_1 - a_2x + a_3x^2 - a_4x^3 + a_5x^4}{c_1 - c_2x + c_3x^2 - c_4x^3}$$

(د) $$P = rA(1+r)^n/[(1+r)^n - 1]$$

٢ – ٥١ سوف تنتج كل من المعادلات المذكورة فى مسألة ٢ – ٥٠ جملة LET طويلة . استعض عن كل جملة ببضع جمل قصيرة متتالية من جمل LET البسيطة .

٢ – ٥٢ اكتب معادلة جبرية تناظر كلامن جمل LET التالية :

(ا) `10 LET F=A+2*B/C↑.5`

(ب) `20 LET F=A+(2*B/C)↑.5`

(جـ) `30 LET F=(A+2)*(B/C)↑.5`

(د) `40 LET F=((A+2)*B/C)↑.5`

(هـ) `50 LET G=P*Q/R*S/T`

٢ – ٥٣ ما هى الصعوبة التى يمكن أن نواجهها عند تنفيذ الجملة :

```
15 LET X=(Y-Z)↑.25
```

٢ – ٥٤ ادرس الجملة التالية :

```
25 LET P=-Q↑4
```

ما هى القيمة التى تعطى المتغير P إذا كانت قيمة Q = 2 ؟

٢ – ٥٥ إدرس الجملة التالية :

```
35 LET P=Q↑4
```

ما هى القيمة التى تعطى للمتغير P إذا كانت قيمة Q = −2 ؟ (قارن النتيجة بإجابة المسألة ٢ – ٥٤ السابقة) .

٢ – ٥٦ اكتب جملة مناسبة أو مجموعة من الجمل لكل من المواقف الموصوفة التالية :

(ا) أدخل قيما عددية للمتغيرات A وB وC وقيما حرفية للمتغيرات M$ و N$. يجب أن تطبع كل البيانات على سطر واحد من سطور النهاية الطرفية .

(ب) ادخل القيم المعطاة للمتغيرات A و N$ و B على سطر واحد والقيم المعطاة للمتغيرات M$ وC على السطر التالى .

(جـ) أدخل القيم العددية للمتغيرات A وB وC والقيم الحرفية للمتغيرات M$ و N$. يجب أن تطبع كل قيمة من هذه القيم فى بداية سطر جديد .

(د) اطبع رسالة تقول :

ENTER VALUES FOR A,B,C,M$ AND N$

ثم أدخل بعد ذلك البيانات المطلوبة على نفس السطر المطبوع عليه الرسالة .

(هـ) اطبع الرسالة المعطاة فى الجزء (د) . ثم بعد ذلك أدخل البيانات المطلوبة على السطر التالى .

(و) اطبع قيم المتغيرات A وB وC و M$ و N$ على سطر واحد مع الفصل بينها بمسافات طبيعية .

(ز) اطبع قيم المتغيرات A وB وC على سطر واحد ، اجعل المسافات بين البنود متقاربة بقدر المستطاع . واسمح لمخرجات متعاقبة أن تبدأ على نفس السطر فوراً بعد قيمة المتغير C .

(ح) اطبع قيم المتغيرات A وB وC و (A + B + C)/3 و (A*B*C)↑(1/3) و (A↑2+B↑2+C↑2)↑.5 كلها على سطر واحد واتبعها بسطر خال ثم سطر ثالث تطبع عليه قيمة M$ قرب الهامش الأيسر وقيمة N$ قرب الهامش الأيمن .

(ط) اطبع القيم العددية للمتغيرات A وB وC كلها على سطر واحد . ويسبق كل عدد عنوان مناسب يصف العدد .

(ى) اطبع قيم المتغيرات M$ ، N$ على سطرين منفصلين يفصل بينهما سطر خال . اسبق قيمة المتغير M$ بالعنوان NAME واسبق قيمة المتغير N$ بالعنوان SOCIAL SECURITY NUMBER حاول أن تجعل المخرجات فى المنتصف ومتقاربه من بعضها بقدر المستطاع .

٢ – ٥٧ بين كيف تظهر البيانات المدخله لكل من المواقف التالية :

(ا)

```
5 INPUT A,B,C
10 INPUT M$,N$
```

حيث A = 0.0000062 ، B = 27.5×10^{-12} ، C = −1000 ، M$ = SHARON ، N$ = GAIL

(ب)

```
20 INPUT P1,P2,T$
```

حيث P1 = −743.08 ، P2 = 0.00987 ، T$ = SUSAN

(جـ)

```
25 INPUT A$,B$,C$
```

حيث A$ = NEW YORK ، B$ = CHICAGO ، C$ = SAN FRANCISCO

(د)

```
15 INPUT P,P$,Q,Q$
```

حيث P = 2,770,543 ، P$ = DECEMBER 29, 1963 ، Q = 48.8×10^{9} ، Q$ = ELEVEN O'CLOCK

٢ – ٥٨ بين كيف تظهر البيانات المخرجة لكل من المواقف التالية :

(ا)

```
100 PRINT A;B;C;P;P1;P2;Q
```

حيث

A = 0.0000062
B = 27.5×10^{-12}
C = −1000
P = 2,770,543
P1 = −743.08
P2 = 0.00987
Q = 48.8×10^{9}

(ب)

```
110 PRINT A,B,C,P,P1,P2,Q
```

حيث قيم المتغيرات هى نفس القيم فى الجزء (ا)

(جـ)

```
120 PRINT A+B*C,P/Q,P1/P2
```

حيث قيم المتغيرات هى نفس القيم فى الجزء (أ)

(د)

```
130 PRINT M$,P$,Q$
```

حيث

M$ = SHARON
P$ = DECEMBER 29, 1963
Q$ = ELEVEN O'CLOCK

٢ – ٥٩ بين كيف توضع التعليقات (أو الملاحظات) فى برنامج بيسك لكل من الحالات التالية :

(ا) أضف عنوان البرنامج AVERAGING OF AIR POLLUTION DATA

(ب) أدخل الملاحظة BEGIN LOOP TO CALCULATE CUMULATIVE SUM

(جـ) أضف التعليق CALCULATE AVERAGE VALUE إلى الجملة :

```
80 LET A=S/N
```

(د) أضف التعليق READ A DATA POINT إلى الجملة التالية :

```
20 INPUT X,T
```

٢ – ٦٠ فيما يلى نبين عدة جمل GO TO تعرف على أى جملة كتبت خطأ .

(ا) `100 GO TO 12`
(ب) `75 GO TO K+1`
(جـ) `30 GO TO 30`
(د) `55 GO TO 400`
(هـ) `20 GO TO "60"`

مســائل للبرمجــة
Programming Problems

٢ – ٦١ جهز خريطة سير عمليات للبرنامج المبين فى مثال ٢ – ٣٠ قارن بين هذه الخريطة وخريطة سير العمليات للمثال ٢ – ٢٦ المبينة فى شكل ٢ – ١ .

٢ – ٦٢ اكتب برنامجاً كاملا مكتوباً بلغة البيسك لكل من المسائل التالية :

(ا) اطبع HELLO فى منتصف السطر .

(ب) اجعل الحاسب يطبع ؟

HI, WHAT'S YOUR NAME?

على سطر . ثم بعد ذلك يدخل المستفيد اسمه أو اسمها فوراً بعد علامة الاستفهام . ثم يترك الحاسب سطرين ويطبع .

WELCOME (name)!

LET'S BE FRIENDS!

على سطرين متعاقبين .

٢ – ٦٣ اكتب برنامجاً كاملا من النوع التخاطبي بلغة البيسك لكل من المسائل التالية :

(ا) يجب أن تقرأ درجات الحرارة لحاسب بالدرجات الفهرنهيتيه ثم تحول إلى درجات مئوية باستخدام الصيغة :

$$°C = \frac{5}{9}(°F - 32)$$

(ب) تحتوى حصالة على عدد n_1 من أنصاف الدولار و n_2 من أرباع الدولار و n_3 من وحدات العشر سنتات و n_4 من وحدات السنتات الخمسة ، n_5 من السنتات . كم عدد النقود فى الحصالة بدلالة الدولارات ؟ (الدولار مائة سنت) .

٢ – ٦٤ ابتدع مخططاً تمهيدياً ثم ارسم خريطة سير العمليات ثم بعد ذلك اكتب برنامجاً كاملا بلغة بيسك لكل من المسائل التالية . اكتب كل برنامج بطريقة تمكنه من تشغيل عدة مجموعات من البيانات المتتالية . وتأكد من أن كل البيانات المخرجة معنونة بوضوح .

(ا) احسب حجم ومساحة الكرة باستخدام الصيغ الرياضية :

$$V = 4\pi r^3/3$$
$$A = 4\pi r^2$$

حيث r هى نصف قطر الكرة .

(ب) يرتبط ضغط وحجم ودرجة الحرارة لكتلة من الهواء بالصيغة الرياضية .

$$PV = 0.37\,m(T + 460)$$

حيث $P =$ الضغط بالرطل لكل بوصة مربعة .

$V =$ الحجم ، بالقدم المكعب ،

$m =$ كتلة الهواء ، بالرطل ،

$T =$ درجة الحرارة بالفرنهيت . °F .

إذا كان إطار العربة يحتوى على 2 قدم مكعب من الهواء منفوخ بضغط 28 رطل لكل بوصه مربعة عند درجة حرارة الحجرة . فكم هى كية الهواء الموجودة بالإطار ؟

(ج) إذا كانت a وb وc تمثل ثلاثة أضلاع للمثلث ، فإن مساحة المثلث :

$$A = \sqrt{s(s-a)(s-b)(s-c)}$$

حيث $s = (a + b + c)/2$ وأيضاً **نصف قطر الدائرة الخارجية :**

$$r_i = A/s$$

ونصف قطر الدائرة الداخلية :

$$r_c = abc/(4A)$$

فاحسب مساحة المثلث ، ومساحة الدائرة الخارجية ، ومساحة الدائرة الداخلية لكل من مجموعات البيانات التالية :

a	11.88	5.55	10.00	13.75	12.00	20.42	7.17	173.67
b	8.06	4.54	10.00	9.89	8.00	27.24	2.97	87.38
c	12.75	7.56	10.00	11.42	12.00	31.59	6.66	139.01

(د) تفرض أن كمية P من الدولارات قد تم استثمارها بنسبة ربح سنوية مقدارها i (معبر عنها بقيمة عشرية) . إذا أعيد استثمار الربح بعد عدد n من السنوات فإن إجمالى كمية النقود F يمكن تحديدها بالمعادلة $F = P(1 + i)^n$ (ويعرف هذا بقانون الربح المركب) .

– إذا استثمرت الكمية 5000 $ بربح مركب سنويا مقداره 6% فكم هى كمية النقود التى يتم تجميعها بعد عشر سنوات ؟

– إذا كان الربح المركب ربع سنوى وليس سنوياً فإن المعادلة يجب تغييرها لتصبح $F = P(1 + i/4)^{4n}$

(هـ) زيادة عدد البكتريا المستنبتة مع الوقت تتناسب مباشرة مع عددها . وعلى ذلك كلما كبر العدد فإن البكتريا سوف يزيد عددها أسرع . يمكن التعبير عن العدد رياضياً كما يلى :

$$P = P_0[1 + 0.0289\,t + (0.0289\,t)^2/2 + (0.0289\,t)^3/6 + (0.0289\,t)^4/24 + \cdots + (0.0289\,t)^n/n!]$$

حيث t = الوقت بالساعات بعد وقت المرجع

P_0 = عدد البكتريا عند وقع المرجع

p = عدد البكتريا عند الوقت t

احسب عامل تضاعف العدد (p/p_0) بعد 2 و5 و 10 و 20 و 50 ساعة من وقت المرجع . اذكر الحدود العشرة الأولى من المتوالية (أى اجعل $n = 9$)

الفصـل ٣

تشغيل برنامج بيسك

Running a BASIC Program

والآن وقد تعلمنا كيفية كتابة برامج بيسك بسيطة ، دعنا نر كيفية إدخال هذه البرامج للحاسب ، وتنقيحها ، والاحتفاظ بها ، وعمل قائمة بها وتنفيذها . سوف تأخذ فى الاعتبار بعض الطرق لاكتشاف بعض أنواع الأخطاء المتعددة التى يمكن حدوثها فى البرامج المكتوبة بطريقة غير صحيحة وكيفية تصحيحها .

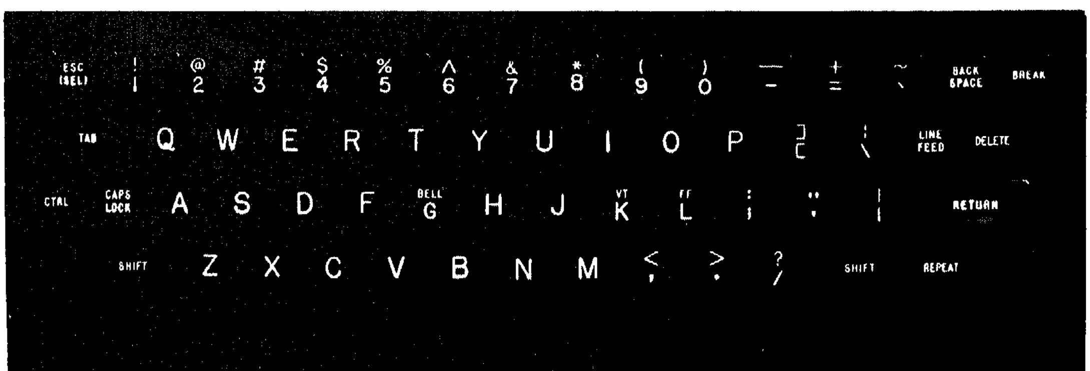
وسوف نوجه انتباهنا فى هذا الفصل تجاه استخدام حاسبات ضخمة فى بيئة مشاركة الزمنية . وسنفترض أن الوحدة الطرفية لإنتاج « نسخة مادية » مشابهة لتلك الموضحة فى شكل ١ــ٣ ، تستخدم للإتصال مع الحاسب المضيف (أى الحاسب الكبير) . واستخدام نهايات طرفية من نوع Video مماثلاً لذلك ، فيما عدا أن المعلومات القديمة ستلف من أعلى الشاشة عندما تتولد المعلومات الجديدة فى أسفل الشاشة . ١

وتشغيل برنامج البيسك على الحاسب الدقيق يماثل ذلك ، إلا أن التفاصيل تختلف نوعاً ما (فى الحقيقة نجد أن طرق تشغيل البيسك مختلفة نوعاً ما على كل الحاسبات بصرف النظر عن حجمها) . ويحتوى الفصل العاشر (انظر قسم ١٠ــ٢ ، ١٠ــ٧) على أمثلة توضح كيف يمكن وضع برامج البيسك فى قائمة وتنقيحها ، وتخزينها ، وتنفيذها على حاسب دقيق .

٣ ــ ١ النهاية الطرفية للمشاركة الزمنية THE TIMESHARING TERMINAL

لقد تعلمنا أنه يجب على المبرمج أن يتعامل مع الحاسب من خلال نهاية طرفية للمشاركة الزمنية (أو كونسول) وذلك عند تشغيلها فى بيئة مشاركة زمنية . يمكن أن تتصل هذه الأجهزة مباشرة بواسطة سلك إلى الحاسب أو يمكن استخدامها للإتصال عن طريق خط تليفون . وفى الحقيقة فإن بعض نهايات المشاركة الزمنية تتضمن مودم مبنى داخله وقرص تليفون . وعلاوة على ذلك ، فإن بعض النهايات الطرفية يمكنها القيام بتخزين البرامج وتنقيحها وتحويلها من وإلى الحاسب المضيف . مثل هذه الأجهزة غالباً ما يطلق عليها نهايات طرفية ذكية (« ذكية ») .

يبين شكل ٣ – ١ صورة قريبة للوحة المفاتيح للنهاية الطرفية الممثلة فى شكل ١ – ٣ . يوجد بعض النهايات الطرفية لها لوحات مفاتيح مماثلة لذلك تماماً إن لم تكن طبق الأصل . وسوف نرجع إلى بعض هذه المفاتيح فى أقسام متعاقبة من هذا الفصل .



شكل ٣ – ١

٣ ــ ٢ التسجيل لدخول النظام LOGGING IN

الخطوة الأولى للاتصال بالحاسب من خلال نهاية طرفية فى المشاركة الزمنية هو إنشاء اتصال بالحاسب . ويتم إنجاز ذلك بواسطة الإجراء Login (أو Logon) . ويمثل الوصف التالى إجراء التسجيل لدخول النظام ، بالرغم من احتمال بعض تغييرات من نسخة بيسك لأخرى(*) .

إذا كان الاتصال سوف يتم من خلال خط تليفونى ، فإن إجراء التسجيل لدخول النظام سوف يبدأ بأن يطلب المستفيد رقم تليفون معين (وذلك بعد توصيل النهاية الطرفية بمنبع الكهرباء) . ثم بعد ذلك يرد الحاسب ويطبع رسالة موجزة يطلب فيها من المستفيد الذى يريد الدخول للنظام أن يصدر login (أى يعرف نفسه وذلك بتحديد رقم مشروع . . . الخ) . ويجب على المستفيد إذن أن يطبع LOGIN (أو LOGON أو ببساطة LOG) ويتبعها بالمعلومات المطلوبة . ويمكن أن يكون من الضرورى أن يطبع المستفيد BASIC ، مشيراً بذلك أنه يرغب فى العمل بلغة بيسك . وليس بأى لغة أخرى .

ويكون الإجراء مماثلا ، أو يمكن أبسط ، إذا كانت النهاية الطرفية موصلة بواسطة سلك بطريقة مباشرة للحاسب ، وبذلك لايتطلب إتصالا بخط تليفونى وفى هذه الحالة عادة يبدأ الإجراء عندما يطبع المستفيد LOGIN يتبعها رقم المشروع الخاص به .

مثال ٣ – ١

يرغب طالب فى جامعة كبيرة فى تشغيل برنامج بيسك من نهاية طرفية فيها قرص تليفونى . وقد حدد المستفيد رقم المشروع 123456 وإجراء login مبين فيما بعد .

١ – يبدأ الطالب فى توصيل النهاية الطرفية بالكهرباء (يوجد مفتاح التشغيل على النهاية الطرفية المبينة فى الشكل ١ – ٣ قرب الناحية اليسرى من لوحة المفاتيح) .

٢ – ثم يدير الطالب قرص التليفون ويطلب الرقم الصحيح ويجيب الحاسب بطباعة

```
PLEASE LOGIN.
```

٣ – يطبع الطالب كلمة LOG بعد النقطة ثم يضغط على مفتاح RETURN ثم بعد ذلك يجيب الحاسب بطباعة

```
JOB 27  TTY42
#
```

٤ – ثم يدخل الطالب بعد ذلك رقم المشروع الخاص به ، 123456 وذلك بعد علامة # ثم يضغط على مفتاح RETURN مرة أخرى . بعد ذلك يطبع الحاسب :

```
08-FEB          THUR  21:11:46
```

مشيراً إلى التاريخ واليوم والوقت بالترتيب .

٥ – ثم تطبع الكلمة (BASIC) بعد ذلك بواسطة الطالب بعد النقطة . وتسبب هذه الكلمة التوصل إلى نظام بيسك من مكتبة الحاسب للغات البرمجة .

٦ . – يطبع الحاسب :

```
NEW OR OLD-->
```

ويجب على الطالب أن يجيب طبقاً لذلك . فإذا رغب الطالب فى التوصل إلى برنامج بيسك قد تم تخزينه فى مكتبة الحاسب فيجب أن يطبع OLD . إلا أن الطالب يرغب فى هذه الحالة إدخال برنامج جديد . ولذلك فيجب أن يطبع كلمة NEW بعد السهم .

(*) الاجراء الموصوف فى هذا الفصل ينطبق بالتحديد على نظام الحاسب DEC system-10 كما هو منفذ حاليا فى جامعة بيتسبرج بالولايات المتحدة الأمريكية وللسهولة تم حذف بعض التفاصيل من النص .

٧ – وأخيراً ، يطبع الحاسب :

NEW FILE NAME-->

ويجيب الطالب بطباعة اسم البرنامج – فى هذه الحالة SAMPLE . (ويتكون الإسم النموذجى من عدد من الحروف يصل إلى ستة حروف تبدأ بحرف هجائى) ويكون الطالب الآن مستعداً للبدء فى طباعة البرنامج .

والشكل ٣ – ٢ يبين قائمة كاملة بإجراء التسجيل لدخول النظام والبنود التى أتم الطالب إدخالها موضوع تحتها خط .

```
PLEASE LOGIN.

.LOG
JOB 27      TTY42
#123456
08-FEB         THUR      21:11:46

.BASIC

NEW OR OLD-->NEW
NEW FILE NAME-->SAMPLE
```

شكل ٣ – ٢

إذا حدث خطأ فى انتقال المعلومات للحاسب أثناء إجراء التسجيل لدخول النظام فسوف تطبع رسالة مناسبة ويطلب من المستفيد إعادة إدخال المعلومات مرة ثانية .

مثال ٣ – ٢

بالإشارة للموقف الذى تم وصفه فى المثال ٣ – ١ ، فلنفرض أن الطالب قد طبع BASIV بدلا من BASIC أثناء إجراء التسجيل لدخول النظام (المفتاح V بجانب المفتاح C فى أغلب لوحات المفاتيح) فإن الحاسب سوف يجيب بطباعة

?BASIV

ويجب على الطالب أن يطبع BASIC بعد النقطة ويستكمل إجراء التسجيل لدخول النظام . إجراء التسجيل لدخول النظام مبين كاملا فى الشكل ٣ – ٣ . مرة ثانية إجابات الطالب موضوع تحتها خط .

```
PLEASE LOGIN.

.LOG
JOB 32      TTY42
#123456
08-FEB         THUR      21:37:07

.BASIV

?BASIV

.BASIC

NEW OR OLD-->NEW
NEW FILE NAME-->SAMPLE
```

شكل ٣ – ٣

```
PITT DEC-1055/A  54A.31 21:11:46

PLEASE LOGIN OR ATTACH.

.LOG
JOB 27 PITT DEC-1055/A  54A.31 TTY42
#115421/160531
PASSWORD:
ALLOCATION REMAINING: 9.8 UNITS
2111     08-FEB            THUR

.R BASIC

NEW OR OLD-->NEW
NEW FILE NAME-->EX3.1
```

شكل ٣ – ٤

تصدر عديد من أنظمة المشاركة الزمنية كلمة سر منفصلة لكل مستفيد ويجب أن تعطى مع رقم المشروع . ويتم إدخال كلمة السر وذلك بكتابتها على لوحة المفاتيح ولكن الحاسب يجعلها غير مرئية وذلك للاحتفاظ بالسرية ولا يسمح للمستفيد بالدخول للنظام إذا لم يعط كلمة السر الصحيحة لرقم مشروع معين .

والشكل ٣ – ٤ يبين قائمة بإجراء التسجيل لدخول نظام يتطلب كلمة السر . وكما سبق أن ذكرنا سيوضع خط تحت إجابات المستفيد .

٣ – ٣ ادخـــال برنـــامج ENTERING A PROGRAM

فور الانتهاء من إجراء التسجيل لدخول النظام يمكن للمستفيد أن يشرع فى كتابة البرنامج الخاص به ، جملة واحدة (سطر واحد) فى كل مرة . ولعمل ذلك يجب على المستفيد أن ينتظر حتى يطبع الحاسب الرمز

فى بداية السطر . (ويعرف ذلك بالتلقين) . بعد ذلك يطبع المستفيد جمل بيسك مبتدئاً برقم الجملة (رقم السطر) . لن ترسل المعلومات المطبوعة لحاسب ، فى هذه الحالة ، حتى يتم الضغط على مفتاح RETURN . وفور عمل ذلك يتحرك رأس الطباعة لبداية السطر التالى ، ويطلب الحاسب الجملة التالية بطباعة الرمز < مرة أخرى .

يستكمل هذا الإجراء حتى يتم الانتهاء من إدخال جميع جمل البرنامج وإرسالها لحاسب وعندما يولد الحاسب الرمز < بعد آخر جملة – فعلى المستفيد أن يجيب بأمر يوضح فيه ماذا يجب عمله بالبرنامج (مثل RUN وLIST وSAVE . . . الخ) . وسوف نناقش هذه الأوامر فيما بعد فى هذا الفصل .

عند كتابة البرنامج لحاسب لا يستلزم أن تدخل التعليمات لحاسب بنفس الترتيب الذى يتم بها التنفيذ . فسوف يعيد الحاسب تنظيم هذه التعليمات وذلك تبعا لتزايد أرقام الجمل فور الانتهاء من إدخال البرنامج كاملا . (تذكر أن ترتيب أرقام الجمل هو الذى يحدد التوالى الذى يتم بناء عليه تنفيذ الجمل) .

مثال ٣ – ٣

يرغب المبرمج الذى أتم إجراء التسجيل لدخول النظام أن يدخل برنامجاً لحاسب . وبسبب السرعة نسى كتابة الجملتين الأوليتين . واكتشف المبرمج خطأه قبل الانتهاء من إدخال البرنامج ، ومع ذلك ، فقد قام بكتابة الجمل المنسية . ذلك مسموح به تماماً ، حيث أن البرنامج سوف يعاد ترتيبه بالترتيب الصحيح بداخل ذاكرة الحاسب .

يبين شكل ٣ – ٥ قائمة بالجمل بنفس الترتيب الذى تم بها إدخال البرنامج .

```
>20 PRINT "RADIUS=";
>30 INPUT R
>40 LET A=P*R↑2
>50 LET C=2*P*R
>60 PRINT "R=";R,"A=";A,"C=";C
>70 GØ TØ 20
>80 END
>5  REM PRØGRAM TØ CALCULATE AREA AND CIRCUMFERENCE ØF A CIRCLE
>10 LET P=3.1415927
>
```

شكل ٣ – ٥

٣ — ٤ تصحيح الأخطاء CORRECTING ERRORS

من غير المعقول عملياً أن يكتب المبرمج برنامجا كاملا ويدخله للحاسب بدون عمل أى خطأ عفوى . ولذلك يجب أن نعرف طريقة تصحيح الأخطاء المكتوبة أو إضافة أو حذف أو تغيير جملة فور الانتهاء من إرسالها للحاسب . فى هذا القسم سوف نرى أنه من السهل إنجاز مثل هذه العمليات فى البيسك .

يمكن حذف الحروف التى طبعت خطأ بالضغط على مفتاح DELETE (على بعض النهايات الطرفية مفتاح RUBOUT) . سوف يلغى الحرف الأحدث عند الضغط على مفتاح DELETE مرة واحدة ، ويسبب الضغط على المفتاح مرتين حذف الحرفين الأكثر حداثة . . . وهكذا . ومع ذلك ، فثل هذا الحذف يجب أن يكون قبل إرسال السطر الذى يحتوى على هذه الاخطاء للحاسب (أى قبل الضغط على مفتاح RETURN) . وبعد إلغاء الحروف التى ترغب فى حذفها يمكن للمبرمج أن يكمل كتابة الحروف الصحيحة .

مثال ٣ — ٤

يرغب المبرمج فى إدخال برنامج البيسك المبين فى شكل ٣ — ٥ للحاسب . ومع ذلك ، فبينما هو يكتب أول جملة من البرنامج فقد كتب U بدون قصد بدلا من I وبذلك ظهر السطر كما يلى :

```
>20 PRU
```

وعند اكتشافه الخطأ ضغط فورا على مفتاح DELETE مرة واحدة : وبعد ذلك أكمل كتابة باقى الجملة .

وسوف يظهر السطر كاملا كما يلى :

```
>20 PRU\U\INT "RADIUS=";
```

لاحظ أن الحرف المحذوف U مبين بين زوج من الشرط العكسية (\\) . وتطبع الشرط العكسية أوتوماتيكياً عند بدء استخدام مفتاح DELETE وعند الانتهاء منه . ومع ذلك ، فمن المهم أن نفهم أن الشرطة العكسية والحروف المحذوفة لن ترسل إلى الحاسب . وعلى ذلك سوف يقوم الحاسب بتفسير السطر المكتوب كما هو مطلوب :

```
20 PRINT "RADIUS=";
```

مثال ٣ — ٥

دعنا نأخذ فى الاعتبار مرة أخرى خطأ الطباعة الذى تمت مناقشته فى المثال السابق . والآن نفترض أن المبرمج لم يلحظ خطأه إلا بعد كتابة عدة حروف قليلة . وبذلك فان السطر المكتوب يظهر :

```
>20 PRUNT "RA
```

عند اكتشاف الخطأ . فيجب على المبرمج أن يضغط على مفتاح DELETE 7 مرات وذلك لحذف كل شئ بدءاً من (ومتضمنا) حرف U ، وبعد ذلك يعيد كتابة باقى الجملة بطريقة صحيحة .

سوف يظهر السطر الأول مكتوباً كالتالى :

```
>20 PRUNT "RA\AR" TNU\INT "RADIUS=";
```

ومرة أخرى نرى أن الحروف المحذوفة محصورة بين زوج من الشرطات العكسية سوف تطبع الشرطتان أوتوماتيكياً عند ابتداء استعمال مفتاح DELETE وعند الانتهاء منه . لاحظ أن الحذف يجرى بطريقة عكسية « أى أن أول حرف يحذف هو الحرف A ، ثم يتبعه الحرف R و . . . الخ . حتى يتم حذف الحرف U . وحيث أن هذه الحروف لم ترسل للحاسب فان السطر سوف يخزن بهذه الطريقة :

```
20 PRINT "RADIUS=";
```

فى بعض الأحيان لا يلحظ الخطأ إلا بعد كتابة السطر كاملا (أو معظمه) فى مثل هذا الموقف يمكن أن يكون حذف حرف فى المرة الواحدة بمفتاح DELETE أو (مفتاح RUBOUT) عملية مرهقة جداً . ويكون الإجراء الأفضل هو حذف السطر كاملا ، والاستعاضة عنه بسطر جديد . يمكن حذف السطر بالضغط على مفتاح ALTMODE أو مفتاح ESCAPE (المسمى ESC فى شكل ٣ – ١) وذلك إذا لم يتم إرسال السطر للحاسب . أما إذا كان قد تم إرسال السطر فيمكن الاستعاضة عنه ببساطة بادخال سطر جديد له نفس رقم الجملة كرقم السطر القديم .

مثال ٣ – ٦

عند إدخال البرنامج المبين فى شكل ٣ – ٥ نفرض أن المبرمج قد كتب

```
>20 PRUNT "RA
```

وبعد ذلك اتضح أنه عمل خطأ (وذلك بطباعة U بدلا من I) . يمكن حذف السطر غير الصحيح بالضغط على مفتاح ESC وإعادة كتابة السطر صحيحاً بعد ذلك .

فى طريقة أخرى يمكنه الضغط على مفتاح RETURN ، وبذلك يدخل الجملة غير صحيحة وغير كاملة للحاسب . ويمكنه بعد ذلك استكمال وكتابة الجملة الصحيحة أى :

```
>20 PRINT "RADIUS=";
```

عند الضغط على مفتاح RETURN مرة أخرى سوف يتم إدخال الجملة للحاسب وبذلك تستبدل الجملة السابقة (غير الصحيحة) التى لها نفس رقم الجملة .

وببساطة يمكن حذف جملة كاملة من برنامج بيسك بكتابة رقم الجملة وبعد ذلك بضغط مفتاح RETURN.

مثال ٣ – ٧

افرض أن البرنامج المبين فى شكل ٣ – ٥ قد تم إدخاله للحاسب وبعد ذلك قرر المبرمج حذف جملة REM فيكتب :

```
>5
```

فقط ويضغط على مفتاح RETURN ، وبذلك تلغى الجملة رقم 5 (جملة REM) .

٣ – ٥ تشغيل برنامج PROCESSING A PROGRAM

يكون البرنامج جاهزاً للتشغيل فور إدخاله للحاسب وتصحيح كل الأخطاء المعروفة . وغالباً ما نريد إعادة كتابة البرنامج (أى عمل قائمة به) وتخزين البرنامج (أى الاحتفاظ به) لاستخدامه فيما بعد ، وبالطبع تنفيذ (أى تشغيل) البرنامج . تجرى هذه العمليات بسهولة بكتابة الكلمات LIST و SAVE و RUN . ويمكن للمبرمج إصدار هذه الأوامر بأى ترتيب يرغب فيه .

```
PITT DEC-1055/A  548.01B 18:21:29

PLEASE LOGIN OR ATTACH.

.LOGIN
JOB 42 PITT DEC-1055/A  54B.01B TTY42
#115421/160531
PASSWORD:
ALLOCATION REMAINING: 9.8 UNITS
1821     15-FEB          THUR

.R BASIC

NEW OR OLD-->NEW
NEW FILE NAME-->CIRCLE

>20 PRUNT\TNU\INT RADIUS
>30 INPUT R
>40 LET A=P*R^2
>50 LET C=2*P*R
>60 PRINT "R=";R,"A=";A,"C=";C
>70 GOTO 20
>80 END
>5  REM PROGRAM TO CALCULATE AREA AND CIRCUMFERENCE OF A CIRCLE
>10 LET P=3.1416\6\5927
>65 PRINT
>20 PRINT "RADIUS=";
>35 IF R=0 THEN 80
>LIST

CIRCLE          18:27          15-FEB

5  REM PROGRAM TO CALCULATE AREA AND CIRCUMFERENCE OF A CIRCLE
10 LET P=3.141593
20 PRINT "RADIUS=";
30 INPUT R
35 IF R=0 THEN 80
40 LET A=P*R^2
50 LET C=2*P*R
60 PRINT "R=";R,"A=";A,"C=";C
65 PRINT
70 GOTO 20
80 END

>SAVE

>RUN

CIRCLE          18:28          15-FEB

RADIUS= ?15
R= 15           A= 706.858     C= 94.2478

RADIUS= ?6.82
R= 6.82         A= 146.123     C= 42.8513

RADIUS= ?37.4
R= 37.4         A= 4394.33     C= 234.991

RADIUS= ?0

TIME:  0.21 SECS.

>BYE
Job 42, USER[115421,160531]  LOGGED OFF TTY42     1829  15-FEB
SAVED ALL FILES (25 BLOCKS)
CPUTIME 0:01   DISK R+W=77+15  CONNECT=8 MIN   UNITS=0.0101
```

شكل ٣ - ٦

مثال ٣ – ٨ مساحة ومحيط دائرة

تم إدخال برنامج لحاسب مشابه للبرنامج المبين فى شكل ٣ – ٥ ويرغب المبرمج بعد ذلك فى طباعة البرنامج وتخزينه وتنفيذه . وعلى ذلك يطبع المبرمج الكلمة LIST بعد الانتهاء من إدخال البرنامج الأصلى (والذى يمكن أن يحتوى على تصحيح أخطاء ، وجمل غير مرتبة ، و . . . الخ) . ويترتب على ذلك إصدار قائمة للجمل بعد تصحيحها وفى الترتيب الصحيح .

فور انتهاء الحاسب من طباعة قائمة كاملة بالبرنامج ، يكتب المبرمج الكلمات SAVE و RUN من أجل تخزين وتشغيل البرنامج على الترتيب . وأخيراً يكتب المبرمج BYE بعد الانتهاء من التشغيل ، وبذلك ينتهى اتصاله بالحاسب .

يبين شكل ٣ – ٦ قائمة بجلسة مشاركة زمنية كاملة . وكما هو معتاد وضعت خطوط تحت المعلومات التى أدخلها المبرمج . لاحظ أن جمل البرنامج الأصلى تحتوى على تصحيحات أخطاء فى الجمل 10 ، 20 وأن الجملة 20 الأولى قد استبدلت بجملة مصححة بعد ذلك . وأيضاً ، نرى أن الجمل قد كتبت بالتسلسل الذى يتم التنفيذ على أساسه .

بمجرد أن يكتب المبرمج LIST نرى عنوان البرنامج (وهو هنا CIRCLE) قد طبع وتبعته قائمة بجمل البرنامج بتسلسل صحيح . مدخلات ومخرجات البيانات اللازمة لتنفيذ البرنامج موضحة بعد أمر RUN . وأخيراً يضع سطور أخيرة تحتوى على معلومات إحصائية تعطى مع إجراء تسجيل الخروج من النظام وذلك نتيجة لأمر BYE . (وسوف يذكر ذلك بتفصيل أكثر فى قسم ٣ – ٦) .

لا يمكن للمبرمج الاستمرار فى الاتصال بالحاسب بعد تسجيل خروجه عن النظام (إلا ، بالطبع ، فى حالة تسجيل دخوله للنظام مرة ثانية) .

ومع ذلك ، فان برنامج CIRCLE سوف يتم تخزينه للاستخدام المتعاقب وذلك نتيجة لأمر SAVE . (يحتمل أن يخزن البرنامج على قرص أو شريط ممغنط وليس فى الذاكرة الأساسية لحاسب ، ولا يهم هذا المبرمج فى شئ) .

إذا أريد تشغيل برنامج سبق تخزينه ، وليس برنامج يدخل لأول مرة ، فيجب على المبرمج أن يكتب OLD بدلا من NEW بعد إجراء تسجيل دخوله للنظام . وسوف يجيب الحاسب بالرسالة :

OLD FILE NAME-->

وبعد ذلك يجب على المبرمج أن يكتب إسم البرنامج المخزن . وابتداء من هذه النقطة يمكن تشغيل البرنامج بالطريقة التى سبق شرحها .

مثال ٣ – ٩

نفرض أننا نرغب فى تنفيذ برنامج CIRCLE الذى سبق مناقشته فى مثال ٣ – ٨ بعد أن تم تخزينه سابقاً . يبين شكل ٣ – ٧ قائمة لجلسة مشاركة زمنية كاملة متضمنة إجراءات التسجيل للدخول والخروج عن النظام . لاحظ أن الأوامر تشبه إلى حد بعيد الأوامر المبينة فى شكل ٣ – ٦ والفرق الوحيد هو استخدام كلمة OLD بدلا من NEW .

من أجل استبدال برنامج قديم بنسخة أحدث تم تنقيحها (مع الاحتفاظ بنفس الإسم السابق) فببساطة يكتب المبرمج REPLACE ويتبعها إسم البرنامج (لاحظ أن ذلك يتسبب فى تدمير النسخة القديمة من البرنامج نظراً لأن الحاسب سوف يكتب عليها النسخة الأحدث) .

تمكننا لغة البيسك أيضاً من تشغيل البرنامج بطرق أخرى . فمثلا يمكننا إلغاء برنامج سبق تخزينه بكتابة UNSAVE ، أو يمكننا الحصول على قائمة بأسماء البرامج المخزنة بكتابة أمر CATALOG . إجمالا ، فان الأوامر المستخدمة لتشغيل برنامج تسمى **أوامر النظام** (**أو أوامر التنقيح**) . وبالرغم من أن أوامر التنقيح المتاحة سوف تختلف من نسخة بيسك لأخرى ، إلا أن كل نسخ اللغة تحتوى على الخصائص الأساسية الموصوفة أعلاه . ملخص لأوامر النظام الشائعة الاستخدام مبينة فى الملحق C .

٣ — ٦ التسجيل للخروج من النظام LOGGING OUT

لقد رأينا أن الإجراء المطلوب لإنهاء الاتصال بالحاسب (أو إجراء تسجيل الخروج عن النظام) تبدأ بكتابة الكلمة GOODBYE (أو ببساطة BYE) . فور إصدار هذا الأمر فان الحاسب سوف يجيب بكتابة ملخص إحصائى للجلسة الحالية للمشاركة الزمنية . وتتضمن الإحصائيات التاريخ وعدد **الملفات** (أى البرامج ومجموعات البيانات) التى تم الاحتفاظ بها وفترة الاتصال (أى الزمن) ووقت الحاسب الفعلى المستخدم ، وغالباً التكلفة . وإجراء تسجيل الخروج عن النظام موضحة فى الأمثلة ٣ — ٨ و ٣ — ٩ (أنظر الشكلين ٣ — ٦ و ٣ — ٧) بجانب المثال ٣ — ١٠ أدناه .

```
PITT DEC-1055/A  54B.01B 20:13:09

PLEASE LOGIN OR ATTACH.

.LOGIN
JOB 62 PITT DEC-1055/A  54B.01B TTY42
#115421/160531
PASSWORD:____
ALLOCATION REMAINING: 9.8 UNITS
2013     15-FEB           THUR

.R BASIC

NEW OR OLD-->OLD
OLD FILE NAME-->CIRCLE

>RUN

CIRCLE          20:14          15-FEB

RADIUS= ?17.45
R= 17.45          A= 956.623     C= 109.642

RADIUS= ?12.7
R= 12.7           A= 506.707     C= 79.7965

RADIUS= ?32.6
R= 32.6           A= 3338.76     C= 204.832

RADIUS= ?0

TIME:  0.20 SECS.

>BYE
JOB 62, USER[115421,160531]  LOGGED OFF TTY42     2015  15-FEB
SAVED ALL FILES (25 BLOCKS)
CPUTIME 0:01    DISK R+W=98+6  CONNECT=3 MIN    UNITS=0.0078
```

شكل ٣ — ٧

مثال ٣ — ١٠

لقد أكمل المبرمج تشغيل البرنامج الخاص به وقد كتب كلمة BYE . وأجاب الحاسب بكتابة الأسطر الثلاثة من المعلومات المبينة فى شكل ٣ — ٨ وبعد ذلك قطع الاتصال بالنهاية الطرفية فور الانتهاء من ذلك .

```
> BYE
JØB 27, USER [115421,160531]  LØGGED ØFF TTY42     2128  8-FEB
SAVED ALL FILES (15 BLØCKS)
CPUTIME 0:07    DISK R+W=856+8    CØNNECT=17 MIN    UNITS=0.0240
```

شكل ٣ — ٨

ينص أول سطر من المخرجات أن المستفيد الذى يحمل رقم حساب 115421 و 160531 قد تم تسجيل خروجه عن النظام من النهاية الطرفية رقم 42 فى الساعة 9:28 مساءاً (أى الساعة 2128) فى يوم 8 فبراير . ويبين السطر الثانى أن عدد 15 « كتلة » من المعلومات قد تم الاحتفاظ بها (حيث أن « كتلة » تحتوى على 640 حرفاً فى هذه النسخة من البيسك بالذات) . ونرى فى السطر الثالث أن المطلوب لتشغيل هذا البرنامج 0.07 ثانية من وقت الحاسب (حقيقة،وقت المشغل المركزى) ، وتم قراءة المعلومات من على جهاز قرص مغناطيسى للتخزين 856 مرة وتم الكتابة على القرص 8 مرات وأن جلسة المشاركة الزمنية بأكلها استغرقت 16 دقيقة بتكلفة 0.0240 وحدة.

٣ — ٧ تحليل الأخطاء ERROR DIAGNOSTICS

غالباً تبقى أخطاء البرمجة بدون اكتشاف حتى تعمل محاولة لتنفيذ البرنامج . وعلى أى حال ، بمجرد إصدار أمر RUN ، سوف يصبح وجود أخطاء معينة واضحاً ، حيث أن مثل هذه الأخطاء سوف تمنع تفسير البرنامج ، أى ، تحويله إلى برنامج بلغة الآلة . وبالتحديد بعض الأخطاء الشائعة من هذا النوع مرجعها متغير غير محدد أو رقم جملة غير محددة أو عدم توازن لأقواس الناحية اليمنى أو اليسرى أو الفشل فى إنهاء البرنامج بجملة END أو . . . الخ . تسمى مثل هذه الأخطاء أخطاء لغوية (أو أخطاء فى تكوين الجملة) .

سوف تولد معظم نسخ بيسك رسالة تحليل عند اكتشاف خطأ لغوى . هذه الرسائل ليست دائماً كاملة وواضحة فى معناها ، ولكن برغم ذلك تساعد فى التعرف على طبيعة ومكان الخطأ .

مثال ٣ — ١١

يبين شكل ٣ — ٩ برنامج بيسك مشابه للبرنامج الممثل فى شكل ٣—٦ ، ما عدا أنه تم تقديم عدة أخطاء لغوية متعمدة . رسائل التحليل التى تولد عند إصدار أمر RUN مبينة بوضوح . لاحظ أنه قد تم اكتشاف الأخطاء الموجودة فى السطور 40 و 80 وإغفال جملة END ومع ذلك ، فالخطأ فى السطر 50 (مشيراً للمتغير B وليس P) لم يتم اكتشافه .

```
EX3.11          22:19          18-FEB

5  REM PRØGRAM TØ CALCULATE AREA AND CIRCUMFERENCE ØF A CIRCLE
10 LET P=3.1415927
20 PRINT "RADIUS=";
30 INPUT R
40 LET A=(P*R↑2
50 LET C=2*B*R
60 PRINT "R=";R,"A=";A,"C=";C
70 PRINT
80 GØ TØ 15

> RUN

EX3.11          22:19          18-FEB

? ILLEGAL FØRMULA IN LINE 40
? UNDEFINED LINE NUMBER 15 IN LINE 80

? NØ END INSTRUCTIØN

TIME: 0.18 SECS.
```

شكل ٣ — ٩

غالباً ما تكون الأخطاء اللغوية وأخطاء الكتابة واضحة عند حدوثها . بينما الأخطاء المنطقية تكون أكثر خداعاً . وهنا ينقل البرنامج بدقة تعليمات المبرمج خالية من أخطاء الكتابة والأخطاء اللغوية ، لكن المبرمج أمد الحاسب بمجموعة من التعليمات الغير صحيحة منطقياً .

أحياناً يكون الخطأ المنطق نتيجة حالة يمكن التعرف عليها بواسطة الحاسب . مثل هذا الموقف ربما ينتج من توليد كمية عددية كبيرة زائدة (يتعدى أكبر رقم مسموح بتخزينه فى الحاسب) أو من محاولة لحساب الجذر التربيعي لرقم سالب ، . . الخ . سوف تولد رسائل التحليل فى مواقف من هذا النوع ، لتجعل من السهل التعرف على الأخطاء وتصحيحها . تسمى رسائل التحليل هذه أخطاء التنفيذ ، تميز بينها وبين أخطاء التفسير التى تم وصفها مسبقاً .

مثال ٣ — ١٢

يبين شكل ٣ — ١٠ برنامج بيسك لحساب الجذور الحقيقية للمعادلة التربيعية .

$$ax^2 + bx + c = 0$$

باستخدام الصيغة التربيعية

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

يخلو البرنامج تماماً من الأخطاء اللغوية . بينما لا يستطيع البرنامج أن يتعامل مع القيم السالبة لكية ($b^2 - 4ac$) (أنظر قاعدة ٣ من القسم ٢ — ٧) . وعلاوة على ذلك ، يحتمل حدوث مصاعب رقمية إذا كان المتغير a له قيمة عددية صغيرة جداً أو قيمة عددية كبيرة جداً (أنظر القاعدة ٥ من القسم ٢ — ١) .

بعد قائمة البرنامج فاننا نرى المخرجات التى تم توليدها للقيم $a = 1$ و $b = 2$ و $c = 3$ ثم للقيم $a = 10^{-30}$ و $b = 10^{10}$ و $c = 10^{36}$. نحصل فى الحالة الأولى على قيمة سالبة للكية ($b^2 - 4ac$) وهذا هو السبب فى رسالتى التحليل الأولى [لاحظ أنه تم استكمال الحسابات باستخدام القيمة المطلقة لكية ($b^2 - 4ac$)] تتسبب المجموعة الثانية من البيانات فى تحديد قيمة كبيرة هائلة للمتغير x_2 وبذلك يتسبب فى رسالة الطفح . (لاحظ أن البيانات المعطاه بواسطة المستخدم موضوع تحتها خط) .

٣ — ٨ تحديد الأخطاء المنطقية LOGICAL DEBUGGING

لقد رأينا أن الأخطاء اللغوية وبعض الأنواع من الأخطاء المنطقية سوف تتسبب فى توليد رسائل تحليلية أثناء ترجمة أو تنفيذ البرنامج . وتكون الأخطاء من هذا النوع سهلة فى اكتشافها وتصحيحها . بينما تكون أخطاء المنطق عادة أكثر صعوبة فى اكتشافها ، حيث المخرجات الناتجة فى برنامج غير صحيح منطقياً يمكن أن تظهر خالية من الأخطاء . وعلاوة على ذلك ، فإن الأخطاء المنطقية غالباً ماتكون صعبة فى اكتشافها حتى عندما يكون وجودها معلوماً (كمثال ، عندما يكون من الواضح أن المخرجات المحسوبة غير صحيحة) . وعلى ذلك فيحتمل أن يلزم الأمر القيام بعمل . « بوليسى » لاكتشاف وتصحيح الأخطاء من هذا النوع . مثل هذا العمل البوليسى يعرف بتحديد **الأخطاء المنطقية** .

```
EX3.12          22:05          22-FEB

10 REM REAL R00TS 0F A QUADRATIC EQUATI0N
20 INPUT A,B,C
30 IF A=0 THEN 90
40 LET D=B↑2-4*A*C
50 LET X1=(-B+D↑.5)/(2*A)
60 LET X2=(-B-D↑.5)/(2*A)
70 PRINT "A=";A,"B=";B,"C=";C,"X1=";X1,"X2=";X2
80 G0 T0 20
90 END

> RUN

EX3.12          22:05          22-FEB


? 1,2,3

% ABS0LUTE VALUE RAISED T0 P0WER IN LINE 50

% ABS0LUTE VALUE RAISED T0 P0WER IN LINE 60
A= 1           B= 2           C= 3           X1= 0.414214  X2=-2.41421
? 1E-30,1E10,1E36

% 0VERFL0W IN LINE 60
A= 1.00000E-30             B= 1.00000E+10             C=
 1.00000E+36  X1=-8.32000E+32              X2=-1.70141E+38
? 0,0,0

TIME: 0.50 SECS.
```

شكل ٣ – ١٠

اكتشاف الأخطاء Detecting Errors

الخطوة الأولى لمجابهة الأخطاء المنطقية هو اكتشاف وجودها وأحياناً يمكن أن يتم إنجاز ذلك باختبار البرنامج الجديد ببيانات سبق معرفة إجاباتها . وإذا لم يتم الحصول على النتائج الصحيحة ، فمعنى ذلك أن البرنامج يحتوى على أخطاء . ومع ذلك ، حتى إذا تم الحصول على النتائج الصحيحة فلا يمكن لنا أن نتأكد تمام التأكد من أن البرنامج خال من الأخطاء ، حيث أن بعض الأخطاء تسبب نتائج غير صحيحة تحت ظروف معينة فقط (ومثال لذلك ، قيم معينة من بيانات الإدخال أو مع أشياء اختيارية فى البرنامج) . ولذلك فيجب أن يجرى على البرنامج الجديد اختبارات دقيقة قبل التأكد أن هذا البرنامج تم تحديد أخطائه المنطقية . ويكون ذلك صحيحاً خصوصاً مع البرامج المعقدة أو البرامج التى سوف يتم استخدامها كثيراً بواسطة آخرين .

وكقاعدة عامة ، يجب إجراء الحسابات يدوياً بمساعدة آلة حاسبة من أجل الحصول على نتائج معروفة . ومع ذلك فى بعض المسائل فإن كمية العمل التى تتضمن إجراء الحسابات يدوياً تكون غير ممكنة . (فالمسألة التى تتطلب بضع ثوانى من وقت حاسب كبير يمكن أن تتطلب عدة أسابيع لحلها يدوياً !) ولذلك فلا يمكن أحياناً تطوير عينة من المسألة لاختبار برنامج جديد . برغم أن تحديد الأخطاء المنطقية لمثل هذه البرامج يمكن أن تكون صعبة فعلا ، فغالباً مايستطيع المبرمج اكتشاف الأخطاء المنطقية بدراسة النتائج المحسوبة بدقة للتأكد من أنها معقولة .

تصحيح الأخطاء Correcting Errors

بمجرد التأكد من أن البرنامج يحتوى على خطأ منطقى . قد يتطلب الأمر شيئاً من الدهاء والبراعة لإيجاد الخطأ . ويجب دائماً أن يبدأ اكتشاف الخطأ بأن يراجع المبرمج كل مجموعة منطقية من الجمل فى البرنامج بمنتهى الدقة . محصن بخلفية وجود خطأ فى مكان ما بالبرنامج

وغالباً مايستطيع المبرمج اكتشاف الخطأ بالدراسة الدقيقة . وإذا لم يكتشف الخطأ ربما يكون من الأصلح ترك البرنامج لفترة من الزمن . ليس أمراً غير عادى لمبرمج مُركز أكثر مما ينبغى إلا يرى خطأ واضحاً من أول مرة .

وإذا لم يتمكن المبرمج من تحديد الخطأ بعد تكرار فحص البرنامج ، فيجب عليه بعد ذلك أن يواصل و يعيد تشغيل البرنامج ، مع طباعة كية كبيرة من المخرجات الوسيطة . ويشار إلى ذلك أحياناً بتتبع البرنامج وغالباً مايتضح مصدر الخطأ فور فحص الحسابات الوسيطة بدقة .

وعندما ينتهى المبرمج من محاولة كل الحيل التى يمكن أن يفكر فيها ولم يجد الخطأ فربما يميل إلى التوهم بوجود خطأ بالماكينة أو خطأ فى عملية الترجمة . بالرغم من ندرة ذلك فن المحتمل حدوث مثل هذا الخطأ . (أحياناً تكون أخطاء الماكينة متقطعة . بينما أخطاء الترجمة تكون متصلة وبذلك تتكرر بنفس الصورة) . ومع ذلك،ففى بعض الحالات يتم فور التوصل إلى الحل النهائى اكتشاف أن أخطاء الترجمة أو أخطاء الماكينة المشتبه فيها إنما ترجع إلى وجود خطأ منطقى .

وأخيراً يجب أن نعرف القارىء حقيقة أن الأخطاء المنطقية لايمكن الهرب منها فى برمجة الحاسب ، إلا أن المبرمج الواعى يجب أن يعمل كل محاولة للتقليل من حدوثها وبذلك فيجب أن يتوقع المبرمج أن كمية معينة من تحديد الأخطاء مطلوبة كجزء من الجهد الكلى فى كتابة برنامج بيسك حقيقى وله معنى .

مثال ٣ - ١٣

مطلوب من طالب كتابة برنامج بيسك لحساب قيمة الصيغة :

$$y=\left(\frac{x-1}{x}\right)+\frac{1}{2}\left(\frac{x-1}{x}\right)^2+\frac{1}{3}\left(\frac{x-1}{x}\right)^3+\frac{1}{4}\left(\frac{x-1}{x}\right)^4+\frac{1}{5}\left(\frac{x-1}{x}\right)^5$$

ولتبسيط عملية البرمجة فقد عرف الطالب متغيراً جديداً u :

$$u = \left(\frac{x-1}{x}\right)$$

وبذلك أصبحت الصيغة الرياضية كالتالى :

$$y = u+\frac{1}{2}u^2+\frac{1}{3}u^3+\frac{1}{4}u^4+\frac{1}{5}u^5$$

شكل ٣ - ١١ يبين برنامج بيسك كامل لحساب هذه الصيغة :

```
10 PRINT "X=";
20 INPUT X
30 LET U=X-1/X
40 LET Y=U+(U/2)^2+(U/3)^3+(U/4)^4+(U/5)^5
50 PRINT "Y=";Y
60 END

>RUN

EX3.13          21:25          05-MAY

X= ?2
Y= 2.20971

TIME:  0.08 SECS.
```

شكل ٣ - ١١

يعلم الطالب أن y يجب أن تأخذ القيمة 0.69 عندما $x=2$ ومع ذلك ، فى النتائج المحسوبة $y = 2.20971$ عندما تكون $x = 2$ ، كما هو مبين فى شكل ٣ – ١١ . وينتهى الطالب إلى أن البرنامج يحتوى على أخطاء منطقية يجب اكتشافها وتصحيحها .

وبالفحص الدقيق للبرنامج يصبح الطالب مدركاً أن الجملة رقم 30 تنتج قيمة $u = 1.5$ عندما تكون $x = 2$ بينما القيمة الصحيحة يجب أن تكون $u = 0.5$ والسبب فى الخطأ هو حذف الأقواس فى الجملة رقم 30 التى يجب أن تقرأ :

30 LET U = (X — 1)/X

وبعد ذلك صحح الطالب البرنامج وأعاد تشغيله للقيمة $x = 2$. جملة PRINT (السطر رقم 50) قد تم تغييرها أيضاً وبذلك يمكن طبع القيمة المحسوبة للمتغير u بجانب القيمة المحسوبة للمتغير y .

```
10 PRINT "X=";
20 INPUT X
30 LET U=(X-1)/X
40 LET Y=U+(U↑2)/2+(U↑3)/3+(U↑4)/4+(U↑5)/5
50 PRINT "U=";U,"Y=";Y
60 END
>RUN

EX3.13          21:29          05-MAY

X= ?2
U= 0.5          Y= 0.688542


TIME: 0.08 SECS.
```

شكل ٣ – ١٢

```
10 PRINT "X=";
20 INPUT X
30 LET U=(X-1)/X
40 LET Y=U+(U/2)↑2+(U/3)↑3+(U/4)↑4+(U/5)↑5
50 PRINT "U=";U,"Y=";Y
60 END
>RUN

EX3.13          21:27          05-MAY

X= ?2
U= 0.5          Y= 0.567384


TIME: 0.07 SECS.
```

شكل ٣ – ١٣

عند تنفيذ البرنامج تم حساب قيمة u الصحيحة ، ولكن قيمة y مازالت غير صحيحة ، كما هو واضح فى شكل ٣ – ١٢ . ولذلك يستنتج الطالب أن البرنامج يحتوى على خطأ إضافى ، موجود فى مكان ما فى الجملة رقم 40 .

وبعد بعض الدراسات الإضافية اكتشف الطالب أن الجملة رقم 40 حقيقة غير صحيحة . ويجب أن تكتب هذه الجملة كما يلى :

40 LET Y=U+(U↑2)/2+(U↑3)/3+(U↑4)/4+(U↑5)/5

(لاحظ أن أزواج الأقواس غير ضرورية نتيجة للتدرج الطبيعى للعمليات) . وبعد ذلك تم تصحيح البرنامج كما هو مبين فى شكل ٣–١٣ . ترى فى نهاية شكل ٣ – ١٣ أن قيمة y المناظرة لقيمة $x = 2$ قد تم حسابها بالصورة الصحيحة والنتيجة $y = 0.688542$.

٣ – ٩ ملاحظـــات ختامية CLOSING REMARKS

نود أن نذكر القارىء مرة ثانية أن برمجة البيسك هى مهارة تشبه إلى حد كبير تعلم العزف على آلة موسيقية ، حيث لا يمكن تعلمها بسهولة بمجرد قراءة كتاب . ولذلك لابد للقارىء أن ينشغل بنشاط فى كتابه برامجه الخاصة وتنقيح ، وتخزينها ووضعها فى قائمة وتنفيذها وتحديد أخطائها . وفى نهاية هذا الفصل يجد القارىء قائمة بالعديد من البرامج المناسبة ، والمقترحة فى مسائل البرمجة . ونهيب بالقارىء أن يقوم بحل العديد من هذه المسائل ما أمكنه ذلك .

بالرغم من أن مادة هذا الفصل تتجه نحو استخدام الحاسبات الضخمة والتى تعمل فى بأسلوب المشاركة الزمنية ، إلا أنها مفيدة لكل المبرمجين المبتدئين بصرف النظر عن البيئة الحسابية الخاصة بهم .

ومع كل فإننا ندعوا مبرمجى الحاسبات الدقيقة بدراسة الأمثلة فى قسمى ١٠ – ٢ ، ١٠ – ٧ بالإضافة إلى المادة الموجودة فى هذا الفصل ، وذلك قبل محاولتهم تشغيل برامج البيسك الخاصة بهم .

اسئلة للمراجعة

Review Questions

٣ - ١ ما هى النهاية الطرفية المركزية (الكونسول أو نضد التشغيل) . وما هى النهاية الطرفية للمشاركة الزمنية (مودم) ؟ وما هى النهاية الطرفية الذكية « البارعة » ؟

٣ - ٢ صف ، بمصطلحات عامة ، إجراءات التسجيل المستخدمة مع البيسك للدخول إلى النظام والخروج منه .

٣ - ٣ ما هو المقصود بالتلقين ؟ وكيف يشار إلى التلقين على النهاية الطرفية عند إدخال أو تشغيل برنامج بيسك ؟

٣ - ٤ كيف يتم نقل سطر من المعلومات تم طباعتها على النهاية الطرفية لحاسب ؟

٣ - ٥ نفرض أن برنامج بيسك تم إدخاله لحاسب ببعض جمل فى أماكن غير صحيحة . فكيف يتم تصحيح مثل هذا الموقف ؟

٣ - ٦ كيف يمكن إلغاء حرف أو أكثر من سطر على النهاية الطرفية ؟ وكيف يمكن إلغاء سطر بالكامل ؟

٣ - ٧ كيف يمكن تغيير جملة غير صحيحة فور ارسالها لحاسب ؟

٣ - ٨ كيف يمكن إلغاء جملة من برنامج بيسك فور الانتهاء من إرسالها لحاسب ؟

٣ - ٩ كيف يمكن استصدار قائمة ببرنامج على النهاية الطرفية ؟

٣ - ١٠ كيف يمكن لبرنامج بيسك أن يخزن على أى وحدة مغناطيسية لاستخدامها فيما بعد ؟

٣ - ١١ كيف ينفذ برنامج البيسك ؟

٣ - ١٢ افرض أنه تم تخزين برنامج بيسك على وحدة تخزين مغناطيسية والمطلوب استرجاعها لتشغيلها مرةأخرى . كيف يمكن التوصل للبرنامج ؟

٣ - ١٣ صف الغرض من كل من أوامر النظام التالية :

BYE, CATALOG, GOODBYE, LIST, NEW, OLD, REPLACE, RUN, SAVE, UNSAVE

٣ - ١٤ ما هو المقصود بالأخطاء اللغوية ؟

٣ - ١٥ كيف تختلف الأخطاء اللغوية (النحوية) والأخطاء المنطقية كل عن الآخر ؟

٣ - ١٦ أذكر بعض الأخطاء اللغوية الشائعة .

٣ - ١٧ أذكر بعض الأخطاء المنطقية الشائعة .

٣ - ١٨ ما هو المقصود من رسائل التحليل وكيف يمكن التمييز بين تحليل أخطاء التفسير وتحليل أخطاء التنفيذ ؟

٣ - ١٩ هل تولد رسائل تحليل رداً على الأخطاء المنطقية ؟

٣ - ٢٠ ما هو المقصود من تحديد الأخطاء منطقياً ؟ أذكر بعض إجراءات تحديد الأخطاء المنطقية الشائعة .

٣ ٢١ بأى شكل يشبه تعلم كتابة برنامج بيسك تعلم الدق على الطبول ؟

مسائل تكميلية

Supplementary Problems

« المسائل » التالية تتعلق بتجميع المعلومات أكثر منها على حل المسائل حقيقة .

٣ - ٢٢ اجعل نفسك ملما بالنهايات الطرفية للمشاركة الزمنية المستخدمة فى مدرستك أو مكتبك .

(أ) أين مفتاح ON/OFF لتوصيل الكهرباء وقطعها ؟

(ب) هل بالوحدة جهاز تليفون بقرص مبنى داخلياً ؟ وإن كان كذلك ، كيف يعمل ؟

(ج) هل يوجد بالوحدة ذكاء مبنى داخلياً ؟

(د) ما هو المفتاح الذى يسبب إلغاء حرف أو أكثر من سطر ؟ إلغاء سطر بالكامل ؟

(هـ) ما هو المفتاح الذى يسبب إرسال سطر للحاسب ؟

(و) هل يمكن للوحدة أن تعمل بأسلوب LOCAL (أى كوحدة مستقلة إذا أصبحت غير متصلة بالحاسب) ؟ وإن كان كذلك ؟ كيف يمكن عمل ذلك ؟

٣ - ٢٣ حدد الإجراءات المستخدمة لتسجيل الدخول للنظام والخروج منه فى مدرستك أو مكتبك . وما هى المعلومات المطلوبة للدخول للنظام تماماً ؟ ما معنى المعلومات الصادرة من الحاسب أثناء إجراءات التسجيل لدخول النظام والخروج منه ؟

٣ - ٢٤ ما هى أوامر نظام بيسك المستخدم فى مدرستك أو مكتبك ؟ اجعل نفسك ملماً بالأوامر التى لم تتم مناقشتها فى هذا الفصل .

٣ - ٢٥ كيف يمكن التمييز بين تحليل أخطاء الترجمة وتحليل أخطاء التنفيذ فى نسخة بيسك المستخدمة فى مدرستك أو مكتبك ؟

٣ - ٢٦ كيف يتم تحديد تكلفة المشاركة الزمنية فى مدرستك أو مكتبك ؟ وما هى التكلفة الحقيقية لوحدة واحده من المشاركة الزمنية ؟ (لاحظ أنه من المهم أن يجاب على هذا السؤال بواسطة الطالب الذى يتلقى توصيل الحاسب بدون مقابل من خلال معهد تعليمى . إن معظم الطلبة ليس لديهم أى فكرة عن التكلفة التجارية المساوية لاستخدام الحاسب !) .

مسائل للبرمجة

Programming Problems

٣ – ٢٧ اتصل بنظام الحاسب ، حدد أسماء أى برامج تم تخزينها تحت رقم الحساب الخاص بك ثم انفصل عن النظام .

٣ – ٢٨ أدخل البرنامج المبين فى شكل ٣ – ٦ لحساب مساحة ومحيط الدائرة تأكد من تصحيح أى خطأ كتابى . اطلب قائمة بالبرنامج بعد أن تم قراءته بواسطة الحاسب . وعند التأكد من أنه صحيح ، نفذ البرنامج عدة مرات باستخدام قيم تراها مناسبة لنصف القطر . تحقق من أن الإجابات المحسوبة صحيحة وذلك بمقارنتها بالنتائج المحسوبة يدوياً لاحظ أن تنهى التنفيذ عند إدخال قيمة نصف القطر صفر) .

٣ – ٢٩ أدخل ، وصحح ، واعمل قائمة ، ونفذ واحتفظ بالبرامج لقليل من المسائل التالية :

(أ) مسألة « أهلا » hello الموصوفة فى ٢ – ٦٢ (أ) .

(ب) مسألة « ما هو إسمك » ؟ ? what's your name الموصوفة فى ٢ – ٦٢ (ب) .

(جـ) مسألة تحويل درجة الحرارة الموصوفة فى ٢ – ٦٣ (أ) .

(د) مسألة الحصالة الموصوفة فى ٢ – ٦٣ (ب) .

(هـ) حسابات حجم ومساحة الكرة الموصوفة فى ٢ – ٦٤ (أ) .

(و) حساب مساحة المثلث ، مساحات الدائرة الداخلية الكبرى والدائرة الخارجية الصغرى كما هو موصوف فى ٢ – ٦٤ (جـ) .

(ز) مسألة الربح المركب ، كما هى موصوفة فى ٢ – ٦٤ (د) .

(ح) حساب نمو البكتيريا المستزرعة كما هو موصوف فى ٢ – ٦٤ (هـ) .

الفصل ٤

التفرع وتكوين حلقات تكرارية
Branching and Looping

كانت مسائل البرمجة التى درسناها حتى الآن من أنواع « آلة جمع » ومعنى هذا أن الحسابات كانت تتم دائماً بترتيب ثابت . ومع ذلك فإن الاستعمالات المتعددة لحاسب الرقمى والجديرة بالملاحظة لا تقع تحت طائلة قدرته على القيام بعدة عمليات حسابية موصوفة فى فترة قصيرة من الوقت ، ولكن فى الواقع قدرته على عمل قرارات منطقية ثم تنفيذ مجموعة من الأوامر المناسبة ، معتمداً على نتائج هذه القرارات يجعل الحاسب مفيداً جداً . دعنا الآن نحول اهتمامنا إلى هذا الموضوع الهام .

لقد تعلمنا أن عمليات التفرع غير المشروط (أى ، تحويل التحكم ، أو « القفز » من جزء فى البرنامج إلى آخر) يمكن أن تتم فى البيسك بواسطة جملة GO TO (انظر قسم ٢ — ١٤) . هناك موقف آخر يبرز دائماً وهو تجويل التحكم إلى مكان من مكانين فى البرنامج ، معتمداً على نتيجة مقارنة بين كيتين مثل هذه العملية تسمى **عملية التفرع المشروط** ، وتسمح باتخاذ قرارات منطقية داخل الحاسب .

هناك عملية أخرى تطلب دائماً فى برنامج الحاسب ، وهى **تكوين حلقات تكرارية** . وهذا يتضمن تكرار بعض الأجزاء من البرنامج إما عدداً معيناً من المرات أو حتى استيفاء شرط معين . يمكن أن يحتوى الجزء المكرر من البرنامج (الحلقة التكرارية) على عملية تفرع مشروطة تقرر إنهاء الحلقة التكرارية أو تنفيذها على الأقل مرة واحدة أخرى . أما إذا تطلب الأمر إعادة تنفيذ الحلقة التكرارية مرة أخرى ، فيتحول التحكم لبداية الحلقة التكرارية . وبذلك فلا داعى لكتابة الأوامر فى الجزء المكرر من البرنامج أكثر من مرة .

سوف نرى فى هذا الفصل كيف يتم القيام بعمليات التفرع والتكرار فى البيسك وهذا يفتح الباب إلى طراز أوسع وأكثر تشويقا . من المسائل المبرمجة .

٤ — ١ المعاملات الرابطة RELATIONAL OPERATORS

يجب أن نجد طريقة للتعبير عن شروط التساوى أو عدم التساوى من أجل القيام بعملية التفرع المشروط فى البيسك ويتم إنجازها من خلال استعمال **المعاملات الرابطة** . وهذه المعاملات هى :

A تساوى B	A = B
A لا تساوى B	A <> B
A أقل من B	A < B
A أقل من أو تساوى B	A < = B
B أكبر من A	B > A
B أكبر من أو تساوى A	B > = A

تستخدم المعاملات الرابطة لتصل بين كيات عددية (أى ، أرقام ، متغيرات أو صيغ رياضية) أو سلاسل من الحروف ، وبذلك تكون الشروط إما مستوفاة أو غير مستوفاة .

مثال ٤ – ١

فيما يلى نبين عدة شروط بها كميات عددية . كل منها ، إما أن يكون مستوفياً أو غير مستوف ، معتمداً فى ذلك على القيمة العددية للمتغيرات .

```
X=27
N<=.001
C>(C1+C2)↑2
A+B<C+D
P<>Q
Z>=X*Y
```

وبذلك يكون الشرط الأخير مستوفياً إذا كانت قيمة Z أكبر من أو تساوى قيمة حاصل ضرب X*Y فيما عدا ذلك يكون الشرط غير مستوف .

تفسر الشروط غير المتساوية التى تحتوى على سلاسل حرفية بطريقة « القادم قبل » أو « القادم بعد » وليس « أقل من » أو « أكبر من » علاوة على ذلك ، فإن الفراغات التى تلى السلسلة الحرفية يتم تجاهلها عند مقارنة السلاسل الحرفية .

مثال ٤ – ٢

نعرض فيما يلى عدة شروط تحتوى سلاسل حرفية . كل شرط إما أن يكون مستوفياً أو غير مستوف ، معتمداً فى ذلك على القيمة الحرفية المعطاة للمتغيرات الشكلية .

```
N$="SMITH"
P$<>Q$
C$<G$
```

سوف يكون الشرط الأول مستوفياً إذا كانت القيمة المعطاة للمتغير N$ هى SMITH . وإلا سيكون الشرط غير مستوف . وسيكون الشرط الثانى مستوفياً إذا كانت القيمة المعطاة للمتغير P$ مختلفة عن القيمة المعطاة للمتغير Q$ من أجل استيفاء الشرط الأخير يجب أن تأتى القيمة المعطاة للمتغير C$ قبل القيمة المعطاة للمتغير G$. فى قائمة الحروف الأبجدية .

٤ – ٢ التفرع المشروط – جملة IF-THEN

CONDITIONAL BRANCHING—THE IF-THEN STATEMENT

تستخدم جملة *IF-THEN* للقيام بعمليات التفرع المشروط . تتكون الجملة من الكلمتين IF و THEN مفصولة بعلاقة ما ، ويتبعها رقم لجملة بعيدة . وعند تنفيذ جملة IF-THEN سوف يتحول التحكم إلى الجملة البعيدة إذا استوفت العلاقة ، وإلا ، فسوف تنفذ الجملة التى تلى جملة IF-THEN. (لاحظ أنه يمكن تحويل التحكم إلى أى جملة بعيدة بداخل البرنامج ، حتى إذا كانت جملة REM) .

مثال ٤ – ٣

مبين فيما يلى جزء من برنامج بيسك يحتوى على جملة IF-THEN

```
15...
...
50 IF I>=100 THEN 80
55 LET I=I+1
60 GO TO 15
...
80...
```

الطريقة التى سوف يتم بها تنفيذ البرنامج متوقفة على العلاقة :

```
I>=100
```

الموجودة فى جملة IF-THEN إذا استوفت العلاقة الشرط (أى إذا كانت قيمة I أكبر من أو تساوى 100)، فإن الجملة رقم 80 سوف تنفذ بعد ذلك ، وإذا كانت العلاقة غير مستوفاة (أى أن قيمة I أقل من 100) فإن الجملة التى سوف تنفذ بعد ذلك هى الجملة رقم 55 .

لاحظ الطريقة التى تستخدم فيها جملة IF-THEN مقترنة بجملة GO TO فى هذا المثال لتكوين حلقة تكرارية .

كثير من نسخ بيسك تسمح باستخدام الكلمات GO TO بدلا من استخدام THEN .

مثال ٤ — ٤

فيما يلى جملة IF-GO TO متضمنة سلاسل حرفية :

```
45 IF N$="SHARON" GO TO 120
```

(يجب أن يكون واضحاً أن استخدام GO TO بدلا من THEN ليس له أى علاقة بنوع العلاقة ، أى إذا كانت كميات عددية أو سلاسل حرفية) .

مثال ٤ — ٥ : جذور معادلة جبرية Roots of an Algebraic Equation

غالباً ما تستخدم الحاسبات لحل المعادلات الجبرية التى لا يمكن حلها بطرق أولية . فلندرس مثلا ، هذه المعادلة :

$$x^5 + 3x^2 - 10 = 0$$

لا يمكن إعادة ترتيب المعادلة فتنتج حلا صحيحاً للمتغير x ولذلك فسوف نحدد الحل بواسطة تكرار إجراء المحاولة والخطأ (أى ، إجراء تكرارى) والتى ينتج عنها تدقيق متوال للحل الذى تم التوصل إليه بالتخمين أولا .

إجراء حسابى Computational Procedure

سوف نبدأ بإعادة ترتيب المعادلة فى الصيغة :

$$x = \sqrt[5]{10 - 3x^2}$$

وإجراؤنا سيكون بتخمين قيمة للمتغير x ، ونعوض بالقيمة فى الطرف الأيمن من المعادلة التى تم إعادة ترتيبها وبذلك نحسب قيمة جديدة للمتغير x . وهذه القيمة الجديدة سنعوض بها فى الطرف الأيمن بعد ذلك ، ونحصل على قيمة أخرى للمتغير x ، وهكذا . وسوف نستمر فى الإجراء إما حتى تصبح القيم المتتالية للمتغير x قريبة قرباً كافياً (أى أن الطريقة **تقاربت**) أو حتى يتم عمل عدد معين من التكرارات (بذلك نمنع استمرار الحسابات بصورة لا تنتهى فى حالة ما إذا كانت النتائج المحسوبة لا تتقارب)

ولنر كيف تعمل الطريقة ، لنفرض أننا اخترنا قيمة أولية للمتغير x وهى 1.0 . وبالتعويض بهذه القيمة فى الطرف الأيمن من المعادلة ، نحصل على :

$$x = \sqrt[5]{10 - 3(1.0)^2} = 1.47577$$

ونعوض بهذه القيمة الجديدة للمتغير x فى المعادلة سوف نحصل على :

$$x = \sqrt[5]{10 - 3(1.47577)^2} = 1.28225$$

وبتكرار هذا الإجراء نحصل على :

$$x = \sqrt[5]{10 - 3(1.28225)^2} = 1.38344$$
$$x = \sqrt[5]{10 - 3(1.38344)^2} = 1.33613$$

وهكذا . لاحظ أن القيم المتتالية للمتغير x تتقارب لقيمة نهائية .

مخطط تمهيدى للبرنامج The Program Outline

من أجل كتابة مخطط تمهيدى لبرنامج بيسك دعنا نعرف الرموز التالية :

X = قيمة المتغير x الذى نعوض به فى الطرف الأيمن من المعادلة .

X1 = القيمة الجديدة المحسوبة للمتغير x .

I = عداد لحساب مرات التكرار (سوف تزاد قيمة I بمقدار واحد فى كل مرة تكرار تالية) .

N = الحد الأقصى المسموح به لعدد مرات التكرار .

وسوف نستمر فى الحسابات حتى يتم الآتى :

(ا) إما أن يصبح الفرق بين قيمتين متتاليتين للمتغير x أقل من 0.00001 ، أو ،

(ب) أن يصل عدد التكرار I لحد الأقصى المسموح به أى للقيمة (N)

والآن يمكننا أن نكتب مخططاً تمهيدياً لبرنامج بيسك كما يلى :

١ – اقرأ قيمتى X و N

٢ – ضع قيمة أولية فى العداد I (ضع به القيمة 1) .

٣ – احسب قيمة المتغير X 1 باستخدام الصيغة الرياضية :

```
X1=(10-3*X↑2)↑.2
```

٤ – اطبع القيم الجديدة المحسوبة للمتغيرين X1 و I (وبطباعة النتائج لكل تكرار بهذه الطريقة فإننا نرى حقيقة ما إذا كانت الحسابات تتقارب أم لا) .

٥ – اختبر قيمة | X — X1 | (أى القيمة المطلقة للقيمتين المتتاليتين للمتغير x لتحدد ما إذا كانت أقل من 0.00001 أم لا :

(ا) إذا كانت | X — X1 | أقل من 0.00001 ، فاذهب للخطوة 7 (اطبع النتائج النهائية) .

(ب) إذا كانت | X — X1 | أكبر من أو تساوى 0.00001 ، فتقدم لخطوة 6 التالية .

٦ – اختبر قيمة I إذا كانت تساوى N (لاحظ أن قيمة I سوف تكون أقل من N فى المراحل الأولى من الحسابات) .

(ا) إذا كانت I تساوى N فاذهب للخطوة 8 (اطبع رسالة تشير إلى أن الحسابات لم تتقارب) .

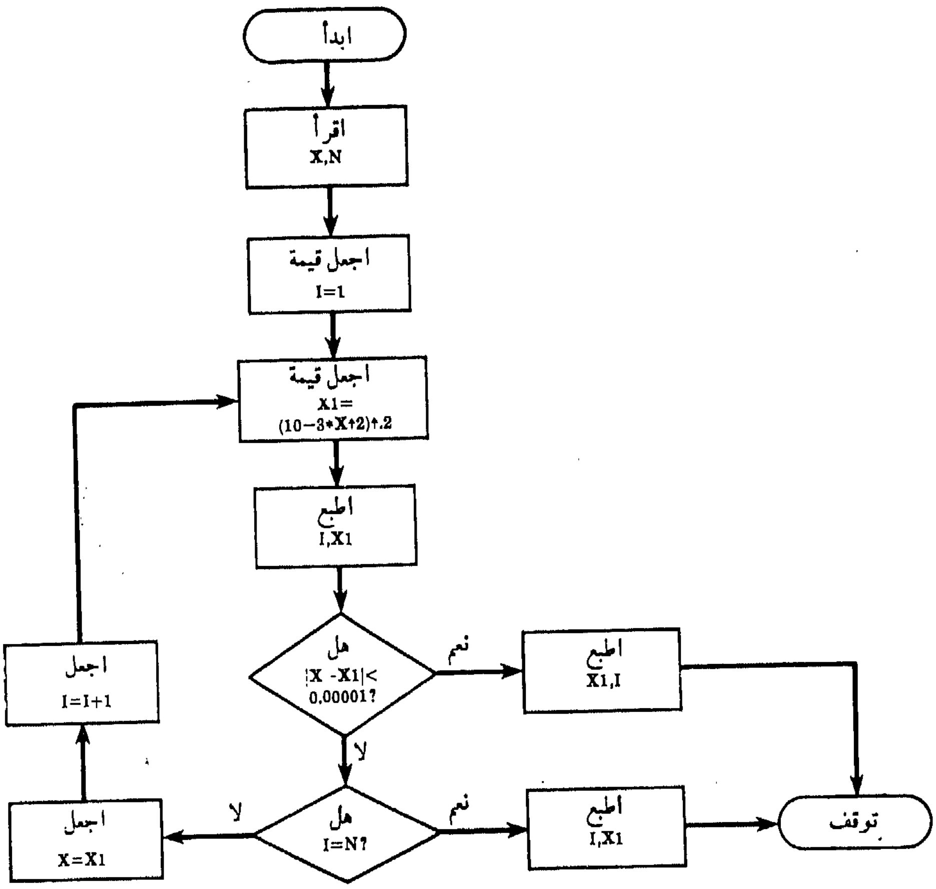
(ب) إذا كانت I أقل من N ، فزد قيمة I بمقدار واحد أى (I = I + 1) اجعل القيمة التى حسبت أخيراً للمتغير X1 تسمى X واذهب مرة أخرى لخطوة 3 ،وبذلك تبدأ التكرار التالى .

٧ – اطبع القيم النهائية للمتغيرين X1 و I ،وبعد ذلك اذهب لخطوة ٩ (توقف) .

٨ – اطبع رسالة تشير إلى أن الحسابات لم تتقارب واتبعها بأحدث القيم للمتغيرين X1 و I .

٩ – توقف .

يبين شكل ٤ – ١ خريطة لسير عمليات إجراء الحسابات .



شكل ٤ – ١

برنامج البيسك The BASIC Program

نرى فى شكل ٤ – ٢ برنامج بيسك كاملا يناظر المخطط التمهيدى وخريطة سير العمليات السابقة . يحتوى البرنامج على جملتى IF فى السطور 60 و 70 . (وبغرض التوضيح فقد تم استخدام كل من الصيغ IF—THEN و IF — GO TO) . لاحظ أن البرنامج يقرأ قيمة مبدئية للمتغير X وقيمة للمتغير N ،ولكن معيار التقارب (أى القيمة 0.00001) قيمة ثابتة بداخل البرنامج ، وكان يمكن أن نعامل معيار التقارب ككمية مدخلة إذا رغبنا فى ذلك . ولاحظ أيضاً ، استخدام الحروف ABS فى السطر 60 . وذلك يشير إلى دالة بيسك المكتبية التى تحدد القيمة المطلقة للكمية (X — X1). (سوف نناقش الدوال المكتبية فى الفصل الخامس ، قسم ٥ – ١) . وأخيراً – نرى أنه كان من الممكن أن نواجه بعض الصعوبات بالبرنامج إذا تم حساب الصيغة الرياضية :

10−3*X↑2

فى السطر 40 وكانت النتيجة سالبة .ومن الممكن أن يتضمن البرنامج اختباراً لمثل هذا الشرط .
لاحظ أن المعلومات المدخلة موضوع تحتها خط

```
>LIST

EX4.5          17:51          03-MAR

10 REM AN ITERATIVE METHOD FOR COMPUTING ROOTS OF AN EQUATION
20 INPUT X,N
25 PRINT
30 LET I=1
40 LET X1=(10-3*X^2)^.2
50 PRINT "I=";I,"X1=";X1
60 IF ABS(X-X1)<.00001 THEN 110
70 IF I=N GOTO 160
80 LET X=X1
90 LET I=I+1
100 GOTO 40
110 PRINT
120 PRINT "THE FINAL ANSWER IS X =";X1
130 PRINT
140 PRINT "NUMBER OF ITERATIONS REQUIRED =";I
150 GOTO 190
160 PRINT
165 PRINT "COMPUTATION HAS NOT CONVERGED AFTER ";I;" ITERATIONS"
170 PRINT
180 PRINT "LAST VALUE OF X =";X1
190 END

>RUN

EX4.5          17:52          03-MAR

?1, 25

I= 1           X1= 1.47577
I= 2           X1= 1.28225
I= 3           X1= 1.38344
I= 4           X1= 1.33613
I= 5           X1= 1.35951
I= 6           X1= 1.34826
I= 7           X1= 1.35375
I= 8           X1= 1.35109
I= 9           X1= 1.35238
I= 10          X1= 1.35175
I= 11          X1= 1.35206
I= 12          X1= 1.35191
I= 13          X1= 1.35198
I= 14          X1= 1.35195
I= 15          X1= 1.35196
I= 16          X1= 1.35195

THE FINAL ANSWER IS X = 1.35195

NUMBER OF ITERATIONS REQUIRED = 16

TIME: 0.36 SECS.
```

شكل ٤ – ٢

يلى قائمة البرنامج المخرجات التى تم توليدها لقيمة التخمين الابتدائية $x = 1.0$ لاحظ أن الحسابات قد تم تقاربها للحل $x = 1.35195$ بعد 16 تكرار . ومن النتائج المطبوعة يمكننا فعلا رؤية أن قيم x المتتالية تتقارب وتتقارب حتى تصل للحل النهائى المتقارب .

يبين شكل ٤ – ٣ النتائج التى تم توليدها إن لم نحصل على التقارب. فى هذه الحالة قد حددنا عدداً أقصى للتكرارات وهى 10 تكرارات فقط $(N = 10)$. وهذا العدد غير كاف للحصول على حل تقاربى بالقيمة الابتدائية $x = 1.0$. يشار بوضوح إلى عدم الوصول إلى التقارب المطلوب بواسطة رسالة مطبوعة

```
>RUN

EX4.5          17:53          03-MAR

?1, 10

I= 1           X1= 1.47577
I= 2           X1= 1.28225
I= 3           X1= 1.38344
I= 4           X1= 1.33613
I= 5           X1= 1.35951
I= 6           X1= 1.34826
I= 7           X1= 1.35375
I= 8           X1= 1.35109
I= 9           X1= 1.35238
I= 10          X1= 1.35175

COMPUTATION HAS NOT CONVERGED AFTER  10  ITERATIONS

LAST VALUE OF X = 1.35175


TIME:  0.26 SECS.
```

شكل ٤ – ٣

٤ — ٣ التفرع المتعدد — جملة ON — GO TO

MULTIPLE BRANCHING—THE ON-GO TO STATEMENT

يمكن القيام بالتفرع المتعدد فى البيسك بواسطة جملة *ON - GOTO* تحتوى هذه الجملة على متغير رقمى أو صيغة رياضية ورقمين أو أكثر لجمل بعيدة . سيتحول التحكم لجملة البعيدة الأولى إذا كانت قيمة المتغير أو الصيغة الرياضية تساوى 1 وإلى الجملة البعيدة الثانية إذا كانت قيمة المتغير أو الصيغة الرياضية تساوى 2 ...إلخ .

مثال ٤ – ٦

مبين فيما يلى جملة ON - GO TO نموذجية :

```
30 ON K GO TO 15,40,25,40,60
```

سوف يتحول التحكم إلى الجملة رقم 15 إذا كانت قيمة K = 1 وإلى الجملة رقم 40 إذا كانت قيمة K = 2 أو K = 4 وإلى الجملة رقم 25 إذا كانت قيمة K = 3 وإلى الجملة رقم 60 إذا كانت قيمة K = 5

إذا كانت قيمة المتغير أو الصيغة الرياضية ليست قيمة صحيحة فسوف يتجاهل الجزء العشرى من الرقم (أى أن الرقم سوف يبتر) .

مثال ٤ – ٧

نفرض أن القيمة التى أعطيت للمتغير K فى المثال ٤ – ٦ هى 3.67 . سوف يتجاهل الجزء 67 . وسوف نعتبر أن قيمة K هى 3 . وبذلك فسوف يتحول التحكم لجملة رقم 25 .

لاحظ أن القيمة الأصلية التى تم إعطاؤها للمتغير K قد تم بترها وليس تقريبها .

تسمح معظم نسخ البيسك باستخدام كلمة THEN بدلا من GO TO

مثال ٤ – ٨

نبين فيما يلي جملة ON — GO TO نموذجية :

```
50 ON A↑2+B↑2 GO TO 20,150,180
```

فى معظم نسخ البيسك يمكن أن تكتب الجملة السابقة كما يلى :

```
50 ON A↑2+B↑2 THEN 20,150,180
```

٤ – ٤ جملة STOP THE STOP STATEMENT

تستخدم جملة *STOP* لإنهاء الحسابات عند أى نقطة من البرنامج . وهى مساوية لجملة GO TO التى تحول التحكم لجملة END. وتتكون الجملة ببساطة من رقم سطر وتتبعها الكلمة STOP

من المهم أن نفهم الاختلاف بين جملتى STOP و END ، يمكن أن تظهر جملة STOP **فى أى مكان من برنامج البيسك ،** ما عدا فى آخر البرنامج . ويمكن أن تظهر أكثر من جملة STOP وعلى العكس فإن جملة END **لا يمكن أن تظهر فى أى مكان ما عدا** فى نهاية البرنامج ، وبذلك فلا يمكن استخدامها أكثر من مرة واحدة . فى البرنامج . (تذكر أن كل برنامج بيسك **يجب** أن ينتهى بجملة END) . مثال ٤ – ٩ التالى يوضح استخدام جملة STOP بجانب جملة ON — GO TO .

مثال ٤ – ٩ حساب قيمة الاستهلاك Calculating Depreciation

دعنا ندرس كيفية حساب قيمة الاستهلاك السنوى لعنصر تتناقص قيمته (مثل ، مبنى ، قطع آلات ، .. إلخ) . توجد ثلاث طرق مختلفة لحساب قيمة الاستهلاك تعرف **بطريقة الخط المستقيم** وطريقة **توازن الانحراف المزدوج** وطريقة **مجموع أرقام السنوات .** نود كتابة برنامج بيسك يسمح لنا أختيار أى من هذه الطرق لكل مجموعة من الحسابات .

طريقة إجراء الحسابات Computational Procedure

سوف تبدأ الحسابات بقراءة القيمة الأصلية (بدون استهلاك) للعنصر ، مدة حياة العنصر (عدد السنوات التى على مداها يستهلك العنصر) ورقم صحيح يشير إلى طريقة الاستهلاك المستخدمة . وسوف تحسب وتطبع قيمة الاستهلاك السنوى .

طريقة الخط المستقيم وهى أسهل طريقة يمكن استخدامها . وفى هذه الطريقة ، تقسم القيمة الأصلية على مدة حياته (العدد الإجمالى للسنوات) . وخارج القسمة سوف يكون الكمية التى يستهلك بها العنصر كل سنة . فمثلا ، إذا كان عنصر ثمنه 8000$ سوف يستهلك على مدى عشر سنوات ، فإن الاستهلاك السنوى سوف يكون 800 $ = 10 ÷ 8000 $ سوف تتناقص قيمة العنصر بمبلغ 800$ كل سنة . لاحظ أن الاستهلاك السنوى هو نفس الاستهلاك كل سنة

عند استخدام **طريقة توازن الانحراف المزدوج** ، فإن قيمة العنصر تتناقص **بنسبة مئوية** ثابتة كل سنة (وعلى ذلك فإن **الكمية الحقيقية** للاستهلاك بالدولار سوف تتغير من سنة لأخرى) . وللحصول على معامل الاستهلاك سوف تقسم ٢ على عمر العنصر . وبضرب هذا المعامل فى قيمة العنصر عند بداية كل سنة **(ليست قيمة العنصر الأصلية)** وذلك للحصول على الاستهلاك السنوى .

ولنفرض مثلا ، أننا نرغب فى تخفيض سعر عنصر ثمنه 8000$ على مدى عشر سنوات ، باستخدام **طريقة توازن الانحراف** المزدوج . سوف يكون معامل الاستهلاك 0.20 = 10 ÷ 2 حيث تكون قيمة الاستهلاك للسنة الأولى عبارة عن 8000$ × 0.20 أى تساوى 1600 $ وقيمة الاستهلاك للسنة الثانية تكون (1600 $ — 8000$) 0.20 أى تساوى 1280 = 6400 $ × 0.20 وقيمة الاستهلاك للسنة الثالثة تكون 1024$ = 5210 $ × 0.20 وهكذا .

فى طريقة **مجموع أرقام السنوات** سوف تتناقص قيمة العنصر بنسبة مئوية مختلفة فى كل سنة سوف يكون معامل الاستهلاك كسراً مقامه مجموع الأرقام من 1 إلى N حيث تمثل N طول حياة العنصر (مثلا ، لحياة مدتها عشر سنوات سوف يكون المقام

55 = 10 + 9 + ... + 3 + 2 + 1) . ولأول سنة سيكون البسط N ،وللسنة الثانية سيكون البسط (1 — N) وللسنة الثالثة سيكون البسط (2 — N) ... وهكذا . ويحصل على الاستهلاك السنوى بضرب معامل الاستهلاك فى القيمة الأصلية للعنصر .

ولنر كيف تعمل طريقة مجموع أرقام السنوات ، سوف يستهلك عنصراً قيمته 8000 $ على مدى عشر سنوات . وسوف يكون الاستهلاك للسنة الأولى 1454.55 $ = 8000 $ × (10-55) . ويكون الاستهلاك للسنة الثانية 1309.09 $ = 8000$ × (9/55) وهكذا .

مخطط تمهيدى للبرنامج The Program Outline

نبدأ بتعريف الرموز التالية :

V = قيمة العنصر .

N = عدد السنوات التى يستهلك فيها العنصر (حياته) .

I = رقم صحيح يشير للطريقة التى تستخدم لحساب الاستهلاك :

(ا) I = 1 تشير إلى حساب الاستهلاك بطريقة الخط المستقيم

(ب) I = 2 تشير لطريقة توازن الانحراف المزدوج .

(جـ) I = 3 تشير لطريقة مجموع أرقام السنوات .

J = عداد يشير إلى السنة التى تدرس حالياً .

D1 و D2 و D3 = الاستهلاك السنوى المحسوب لكل من الطرق الثلاث .

وسوف يتبع برنامج البيسك الخاص بنا المخطط التمهيدى الممثل أدناه .

١ – اقرأ V و N و I

٢ – اطبع رسالة تشير إلى الطريقة المستخدمة لحساب الاستهلاك

٣ – ضع صفراً فى العدد J

٤ – احسب D1 = V/N

$$F1=\frac{V}{N*(N+1)/2}$$

(تستخدم F1 فى طريقة مجموع أرقام السنوات . لاحظ أن مجموع الأرقام N ...... + 3 + 2 + 1 مساوياً (N*(N + 1)/2.) .

٥ – أضف 1 للعداد J (أى J = J + 1)

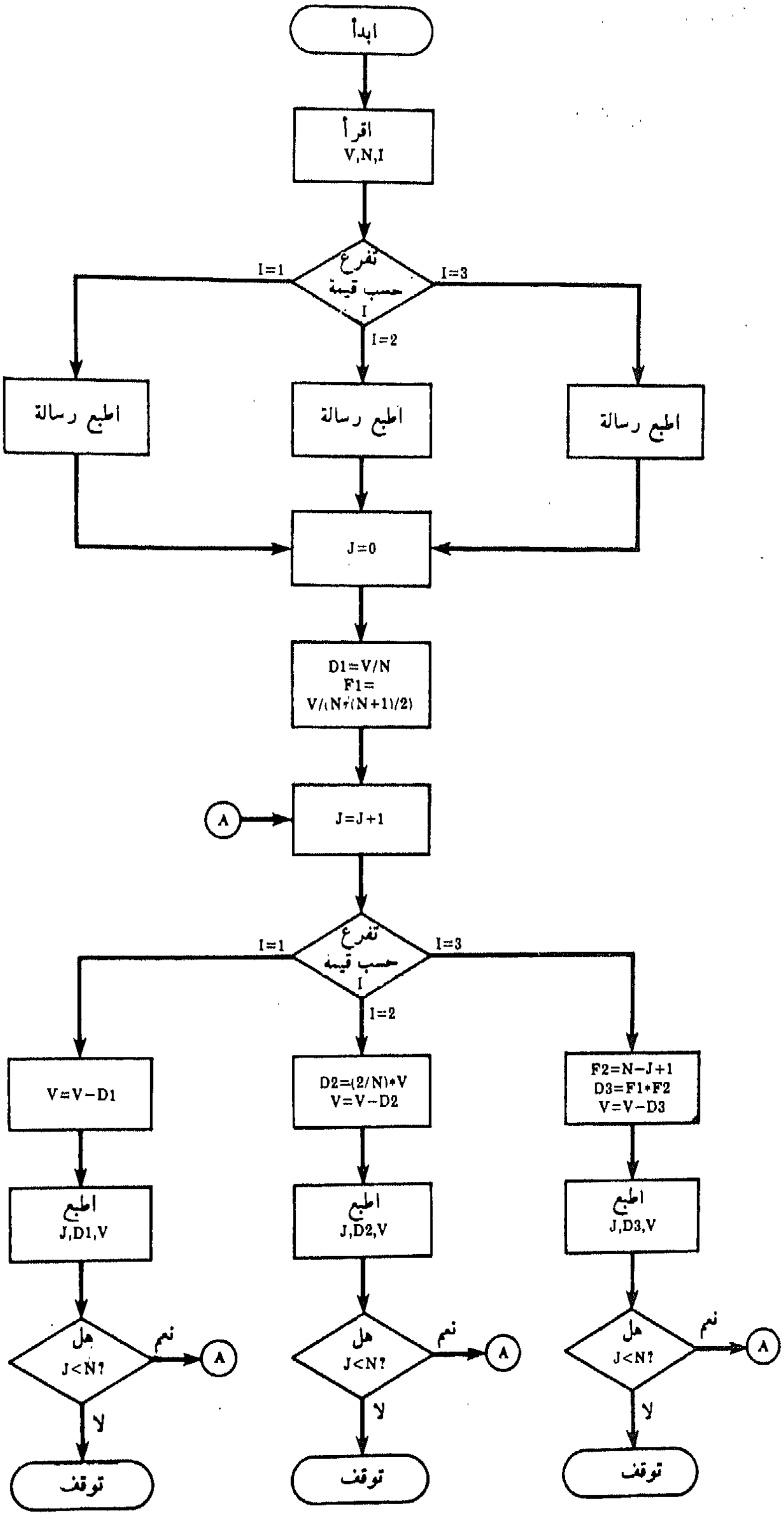
٦ – احسب الاستهلاك السنوى والقيمة الجديدة للعنصر بالطريقة المناسبة واطبع النتائج .

(ا) إذا كانت I = 1 فنفذ ما يلى

(i) احسب V = V — D1

(ii) اطبع J, D1, V

(iii) إذا كانت J < N اذهب لخطوة ٥ وإلا اوقف البرنامج .



ابدأ
اقرأ
V,N,I
تفرع حسب قيمة I
I=1
I=2
I=3
اطبع رسالة
اطبع رسالة
اطبع رسالة
J=0
D1=V/N
F1=
V/(N*(N+1)/2)
J=J+1
A
تفرع حسب قيمة I
I=1
I=2
I=3
V=V-D1
D2=(2/N)*V
V=V-D2
F2=N-J+1
D3=F1*F2
V=V-D3
اطبع
J,D1,V
اطبع
J,D2,V
اطبع
J,D3,V
هل
J<N?
نعم
لا
توقف

شكل ٤ - ٤

```
10 REM COMPUTATION OF DEPRECIATION BY THREE DIFFERENT METHODS
20 PRINT "V=";
30 INPUT V
40 PRINT "N=";
50 INPUT N
60 PRINT "I=";
70 INPUT I
80 PRINT
90 ON I GOTO 100,120,140
100 PRINT "STRAIGHT-LINE METHOD"
110 GOTO 150
120 PRINT "DOUBLE DECLINING BALANCE METHOD"
130 GOTO 150
140 PRINT "SUM-OF-THE-YEARS'-DIGITS METHOD"
150 PRINT
160 PRINT "END OF YEAR","DEPRECIATION","CURRENT VALUE"
170 LET J=0
180 LET D1=V/N
190 LET F1=V/(N*(N+1)/2)
200 LET J=J+1
210 ON I GOTO 300,400,500
215
300 REM STRAIGHT-LINE METHOD
305
310 LET V=V-D1
320 PRINT J,D1,V
330 IF J<N THEN 200
340 STOP
345
400 REM DOUBLE DECLINING BALANCE METHOD
405
410 LET D2=(2/N)*V
420 LET V=V-D2
430 PRINT J,D2,V
440 IF J<N THEN 200
450 STOP
455
500 REM SUM-OF-THE-YEARS'-DIGITS METHOD
505
510 LET F2=N-J+1
520 LET D3=F1*F2
530 LET V=V-D3
540 PRINT J,D3,V
550 IF J<N THEN 200
560 END
```

شكل ٤ – ٥

(ب) إذا كانت $I = 2$ فنفذ ما يلي :

(i) احسب $D2 = (2/N) * V$

$V = V - D2$

(ii) اطبع $J, D2, V$

(iii) إذا كانت $J < N$ اذهب للخطوة 5 ، وإلا أوقف البرنامج .

(جـ) إذا كانت I = 3 فنفذ ما يلى :

(i) احسب $F2 = N - J + 1$

$D3 = F1 * F2$

$V = V - D3$

(ii) اطبع V و D3 و J

(iii) إذا كانت $J < N$ اذهب للخطوة 5 ، وإلا أوقف البرنامج .

يبين الشكل ٤ – ٤ خريطة سير العمليات المناظرة لذلك .

برنامج بيسك The BASIC Program

نرى فى شكل ٤ – ٥ برنامج بيسك كامل للقيام بعمل الحسابات . لاحظ أن البرنامج يحتوى على جملتى ON-GO TO فى السطر 90 والسطر 210 (وكان يمكننا استعمال ON-THEN بنفس السهولة ، وسيكون المنطق متطابقاً) . تمدنا كل من هذه الجمل بشرط له ثلاثة تفرعات فى هذه المسألة . وسنرى أيضاً فائدة جملة STOP فى السطر 340 والسطر 450 وكان يمكننا استخدام GO TO كبديل وكتابة GO TO 550 فى مكان جمل STOP

```
V= ?8000
N= ?10
I= ?1

STRAIGHT-LINE METHOD

END OF YEAR    DEPRECIATION   CURRENT VALUE
 1              800            7200
 2              800            6400
 3              800            5600
 4              800            4800
 5              800            4000
 6              800            3200
 7              800            2400
 8              800            1600
 9              800            800
 10             800            0

TIME:   0.45 SECS.
```

(أ)

```
V= ?8000
N= ?10
I= ?2

DOUBLE DECLINING BALANCE METHOD

END OF YEAR    DEPRECIATION   CURRENT VALUE
 1              1600           6400
 2              1280           5120
 3              1024           4096
 4              819.2          3276.8
 5              655.36         2621.44
 6              524.288        2097.15
 7              419.43         1677.72
 8              335.544        1342.18
 9              268.435        1073.74
 10             214.748        858.993

TIME:   0.43 SECS.
```

(ب)

```
V= ?8000
N= ?10
I= ?3

SUM-OF-THE-YEARS'-DIGITS METHOD

END OF YEAR    DEPRECIATION   CURRENT VALUE
 1              1454.55        6545.45
 2              1309.09        5236.36
 3              1163.64        4072.73
 4              1018.18        3054.55
 5              872.727        2181.82
 6              727.273        1454.55
 7              581.818        872.727
 8              436.364        436.364
 9              290.909        145.455
 10             145.455        2.47955E-5

TIME:   0.31 SECS.
```

(جـ)

شكل ٤ – ٦

تبين الأشكال ٤ – ٦ (ا) ، (ب) ، (جـ) المخرجات التى تم الحصول عليها بطريقة الخط المستقيم وطريقة توازن الانحراف المزدوج وطريقة جمع أرقام السنوات بالترتيب . ونحن نحسب استهلاك عنصر قيمته المبدئية 8000 $ فى كل حالة وفترة حياته التى يستهلك فيها هى عشر سنوات .

لاحظ أن الطريقتين الأخيرتين تنتجان استهلاكاً سنوياً كبيراً خلال السنوات الأولى ، واستهلاكاً سنوياً صغيراً جداً فى السنوات القليلة الأخيرة من حياة العنصر . ونرى أيضاً ، أن العنصر تكون قيمته صفراً عند نهاية عمره وذلك عند استخدام الطريقة الأولى والأخيرة ، ولكن تبقى له قيمة محددة عند استخدام طريقة توازن الانحراف المزدوج . (والرقم 2.47955E-5 بدلا من صفر فى شكل ٤ – ٦ (جـ) ناتج عن تقريب الأعداد) .

٤ ـــ ٥ تكوين الحلقات التكرارية ـــ جملة FOR-TO

BUILDING A LOOP—THE FOR-TO STATEMENT

لقد رأينا فعلا أنه يمكننا بناء حلقات تكرارية فى البيسك باستخدام جملة IF-THEN وجملة GO TO. ويكون ذلك مناسباً إذا لم يكن معلوماً مسبقاً كم عدد مرات التكرار . ومع ذلك ، فغالباً ما يكون لدينا معرفة مسبقة بعدد مرات تنفيذ الحلقة التكرارية . ويمكن بناء الحلقة التكرارية تحت هذه الظروف ببساطة وذلك باستخدام جملتى FOR-TO و NEXT .

تحدد جملة *FOR-TO* عدد مرات تنفيذ الحلقة التكرارية . ويجب أن تكون أول جملة فى الحلقة التكرارية وتتضمن جملة FOR-TO متغيراً عادياً (بدون دليل) ويجب أن يكون متغيراً رقمياً يسمى **المتغير** الجارى وتتغير قيمته بعد كل مرة تنفذ فيها الحلقة التكرارية وتحدد عدد مرات التنفيذ بتوصيف قيمة مبدئية وقيمة نهائية للمتغير الجارى .

مثال ٤ – ١٠

تبين الجملة التالية جملة نموذجية لـ FOR-TO

```
50 FOR I=1 TO 10
```

والمتغير I فى هذا المثال هو المتغير الجارى . وسوف تحدد قيمة I بواحد فى أول مرة تنفذ بها الحلقة التكرارية . وسوف تزاد قيمة I بالوحدة فى كل مرة تكرر فيها الحلقة التكرارية ، حتى تصل I إلى قيمتها النهائية وهى 10 فى آخر مرة تنفذ فيها الحلقة التكرارية . وسوف ينتهى التنفيذ طالما تتعدى قيمة I القيمة النهائية وهى 10 . وبذلك فإن الحلقة التكرارية المعرفة أعلاه بجملة FOR-TO سوف تنفذ 10 مرات .

وسوف تزاد دائماً قيمة المتغير الجارى بالوحدة إذا لم تحتو جملة FOR-TO أى أوامر عكس ذلك . ومع ذلك ، فيمكننا زيادة المتغير الجارى بقيمة أخرى غير الواحد إذا رغبنا فى ذلك . ويمكننا عمل ذلك بإضافة عبارة STEP إلى جملة FOR-TO كما هو موضح فى المثال التالى .

مثال ٤ – ١١

نفرض أننا نريد تنفيذ حلقة تكرارية 50 مرة ، والمطلوب زيادة المتغير الجارى بمقدار وحدتين . بعد كل تنفيذ متعاقب . يمكننا كتابة :

```
75 FOR J=1 TO 99 STEP 2
```

وبذلك تحدد قيمة المتغير الجارى J بالقيمة 1 أثناء المرور الأول والقيمة 3 أثناء المرور الثانى ، والقيمة 5 أثناء المرور الثالث ، ..إلخ . حتى تأخذ J القيمة 99 أثناء المرور رقم 50 (الأخير) .

لا يلزم أن يتقيد المتغير الجارى بقيمة صحيحة موجبة ، بل يمكن أن يأخذ قيما سالبة أو عشرية إذا رغبنا فى ذلك . علاوة على ذلك ، فيمكن للمتغير الجارى أن يتناقص لكل تنفيذ متعاقب لحلقة التكرارية . (ويمكن إنجاز ذلك بتوصيف كية سالبة فى عبارة STEP) وأخيراً يمكن التعبير عن القيمة المبدئية والنهائية وقيمة STEP التى تعطى للمتغير الجارى بمتغيرات أو صيغ رياضية بجانب الأرقام .

مثال ٤ – ١٢

مبين فيما يلى توضيحات لجمل FOR-TO صالحة :

```
30 FOR X=-1.5 TO 2.7 STEP 0.1
15 FOR I=N TO 0 STEP -1
55 FOR K=N1 TO N2 STEP N3
80 FOR F=A/2 TO (B+C)↑2 STEP K+1
```

تسمح بعض نسخ البيسك باستخدام الكلمة BY بدلا من STEP .

مثال ٤ – ١٣

يمكن أيضاً كتابة جملة FOR-TO المبينة فى مثال ٤ – ١١ فى بعض نسخ بيسك .

```
75 FOR J=1 TO 99 BY 2
```

٤ – ٦ انهاء حلقة تكرارية – جملة NEXT

CLOSING A LOOP—THE NEXT STATEMENT

وكما تبدأ دائماً الحلقة التكرارية بجملة FOR-TO ، فإنها دائماً تنتهى بجملة *NEXT* . وتتكون الحلقة التكرارية الكاملة من كل الجمل المتضمنة بين جملتى FOR-TO و NEXT .

تتكون جملة NEXT من رقم الجملة تتبعها الكلمة الدالة NEXT ، ثم يتبعها اسم المتغير الجارى . يجب أن يكون المتغير الجارى هو نفسه المتغير الجارى الذى يظهر فى جملة FOR-TO المناظرة .

مثال ٤ – ١٤

فيما يلى هيكل حلقة تكرارية بنيت باستخدام جملتى FOR-TO و NEXT .

```
50 FOR I=1 TO 10
...
90 NEXT I
```

وسوف تتكون الحلقة التكرارية من كل الجمل ابتداء من الجملة رقم 50 إلى الجملة رقم 90 وسوف يتم تنفيذها 10 مرات .

يجب أن نحتفظ بعدة قواعد عند بناء حلقة تكرارية FOR-TO....NEXT هذه القواعد تتلخص فيما يلى :

١ – يمكن أن يظهر المتغير الجارى فى أى جملة داخل الحلقة التكرارية ولكن لا يمكن أن تتغير قيمته .

٢ – إذا كانت القيمة المبدئية للمتغير الجارى مساوية للقيمة النهائية وقيمة حجم الخطوة ليست صفراً ، فإن الحلقة التكرارية سوف تنفذ مرة واحدة .

٣ – لن تنفذ الحلقة التكرارية إطلاقاً تحت الشروط الثلاثة التالية :

(ا) إذا كانت قيم المتغير الجارى المبدئية والنهائية متساوية وكان حجم الخطوة مساوياً لصفر .

(ب) إذا كانت القيمة النهائية للمتغير الجارى أقل من القيمة المبدئية وكان حجم الخطوة موجبا .

(جـ) إذا كانت القيمة النهائية للمتغير الجارى أكبر من القيمة الأصلية وكان حجم الخطوة سالباً .

(عادة تحدث الشروط الموصوفة فى القاعدتين ٢ و ٣ كخطأ غير مقصود) .

٤ – يمكن تحويل التحكم إلى خارج الحلقة التكرارية ولكن ليس إلى داخلها . (يمكن إنجاز تحويل التحكم إلى خارج الحلقة التكرارية بجملة GO TO أو جملة IF-THEN أو جملة ON-GO TO)

مثال ٤ – ١٥

ادرس هيكل الحلقة التكرارية المبينة فيما يلى :

```
120 FOR X=0 TO 0.5 STEP 0.01
        ...
165     LET Z=X+Y
170     IF Z>Z1 THEN 250
        ...
195 NEXT X
    ...
250 PRINT X,Y,Z
```

يوضح هذا المثال استخدام المتغير الجارى (X) بداخل الحلقة التكرارية (بالتحديد فى الجملة 165) . ونرى أيضاً أن الجملة 170 تحول التحكم إلى خارج الحلقة التكرارية إذا كانت قيمة Z أكبر من القيمة Z1 .وأخيراً ، نرى المتغير الجارى (X) الذى يظهر فى جملة NEXT هو نفسه المتغير الجارى فى جملة FOR-TO كما هو مطلوب .

لاحظ أن الجمل بداخل الحلقة التكرارية ، بين جملتى FOR-TO و NEXT تم إدخالها قليلا لناحية اليمين . ليس هذا مطلوب ولكنه خبرة جيدة فى البرمجة ، حيث تسمح زحزحة الجملة بداخل الحلقة التكرارية لسهولة التعرف عليها .

غالباً يستخدم هيكل الحلقة التكرارية FOR-TO....NEXT فى أنواع عديدة ومختلفة من المسائل . فيما يلى مثال لمسألة نموذجية تتضمن استخدام مثل هذه الحلقة التكرارية .

مثال ٤ – ١٦ حساب متوسط بيانات تلوث الهواء Averaging of Air Pollution Data

يمكن التعبير عن مستوى تلوث الهواء بدلالة نوعية الهواء ، أى ، عدد الدقائق الموجودة فى السنتيمتر المكعب من الهواء الملوث . وعلى ذلك كلما زادت قيمة نوعية الهواء ، كلما ارتفع مستوى التلوث . قياس نوعية الهواء يعمل عدة مرات فى اليوم فى معظم المدن الكبرى .

ولنفرض أننا أعطينا جدولا يحتوى على عدد N من قياسات نوعية الهواء عند أوقات مختلفة خلال اليوم ، كما هو مبين فى الجدول ٤ – ١ (لاحظ الوقت المعطى بدورات 24 ساعة أى 13.00 تشير إلى 1.00 مساء . ولاحظ أيضاً أن 20 = N فى هذا المثال). نود حساب قيمة متوسطة للمتوسط الحسابى لنوعية الهواء للمدة بأكملها . ويمكن إنجاز ذلك أولا بحساب قيمة متوسطة لكل فترة زمنية (أى الفترة الزمنية ما بين قراءتين متعاقبتين) ، ثم بعد ذلك حساب متوسط كامل مرجح من المتوسطات الفردية .

جدول ٤ – ١

الوقت	نوعية الهواء	الوقت	نوعية الهواء
0:00	12.2	13:00	46.6
2:00	12.5	14:00	43.1
1.00	11.9	15:00	39.2
6:00	13.5	16:00	44.7
7:00	22.4	17:00	62.9
8:00	31.4	18:00	88.0
9:00	57.7	19:00	71.4
10:00	84.4	20:00	59.0
11:00	68.0	22:00	43.5
12:00	51.6	24:00	28.7

طريقة إجراء الحسابات Computational Procedure

دعنا أولا نقدم الرموز التالية :

T1 = الوقت عند بدء الفترة الزمنية المعطاة .

T2 = الوقت عند انتهاء الفترة الزمنية المعطاة .

Q1 = نوعية الهواء عند بدء الفترة الزمنية المعطاة .

Q2 = نوعية الهواء عند انتهاء الفترة الزمنية المعطاة .

Q3 = متوسط نوعية الهواء خلال الفترة الزمنية المعطاة .

وسوف نجرى الحسابات كما يلى :

١ – حساب متوسط نوعية الهواء خلال كل فترة زمنية باستخدام الصيغة الرياضية :

$$Q3 = (Q1 + Q2)/2$$

٢ – ضرب كل متوسط لنوعية الهواء بالفترة الزمنية المناظرة (T2 — T1) وتجميع كل حواصل الضرب ، أى :

$$S=[Q3*(T2-T1)]_1+[Q3*(T2-T1)]_2+\cdots+[Q3*(T2-T1)]_{N-1}$$

تشير S فى الصيغة الرياضية السابقة إلى مجموع حواصل الضرب الفردية وتشير الأدلة 1 و 2 و . . . و N — 1 إلى الفترات الزمنية العديدة . (لاحظ أن عدد الفترات الزمنية سوف تكون N — 1 ، حيث يوجد N قيمة لكل من Q و T) .

٣ – اقسم هذا المجموع على الفترة الزمنية الشاملة (زمن آخر قراءة مطروح منه الزمن عند أول قراءة) وذلك للحصول على قيمة متوسط الوقت .

$$A = S/(T9 - T0)$$

حيث تشير A لمتوسط زمن نوعية الهواء ، وتشير كل من T0 و T9 إلى أول قراءة وزمن آخر قراءة بالترتيب .

(وسوف يتعرف القارئ الذى سبق ودرس حساباً عددياً على هذه المسألة كتمرين أولى فى التكامل العددى . وتعرف الطريقة بقاعدة شبه المنحرف) .

المخطط التمهيدى للبرنامج The Program Outline

يمكننا الآن كتابة المخطط التمهيدى المفصل التالى لإجراء الحسابات .

١ – اقرأ N

٢ – اجعل قيمة S مساوية للصفر قبل قراءة أى من البيانات الحقيقية .

٣ – اقرأ أول مجموعة من البيانات (أى القيم المبدئية للمتغيرين T1 و Q1) .

٤ – اجعل قيمة T0 مساوية لقيمة T1 (هذه طريقة من طرق إعطاء علامة لزمن أول قراءة . سوف نحتاج هذه القيمة فى الخطوة السابعة التالية) .

٥ – نفذ الحسابات التالية عدد (N — 1) من المرات :

(ا) اقرأ T2 و Q2

(ب) احسب Q3

(جـ) اجمع حاصل ضرب Q3*(T2 — T1) على قيمة S .

(د) اجعل قيمة T2 مساوية T1 و Q2 مساوية Q1 ، للتجهيز لفترة الزمن التالية . (بمعنى آخر ، تصبح قيم T و Q عند نهاية فترة زمنية معطاة هى قيم T و Q عند بداية الفترة الزمنية التالية) .

٦ – اجعل قيمة T9 مساوية لآخر قيمة T2 .

٧ – احسب A .

٨ – اطبع A والفترة الزمنية الشاملة ، (T9 — T0)

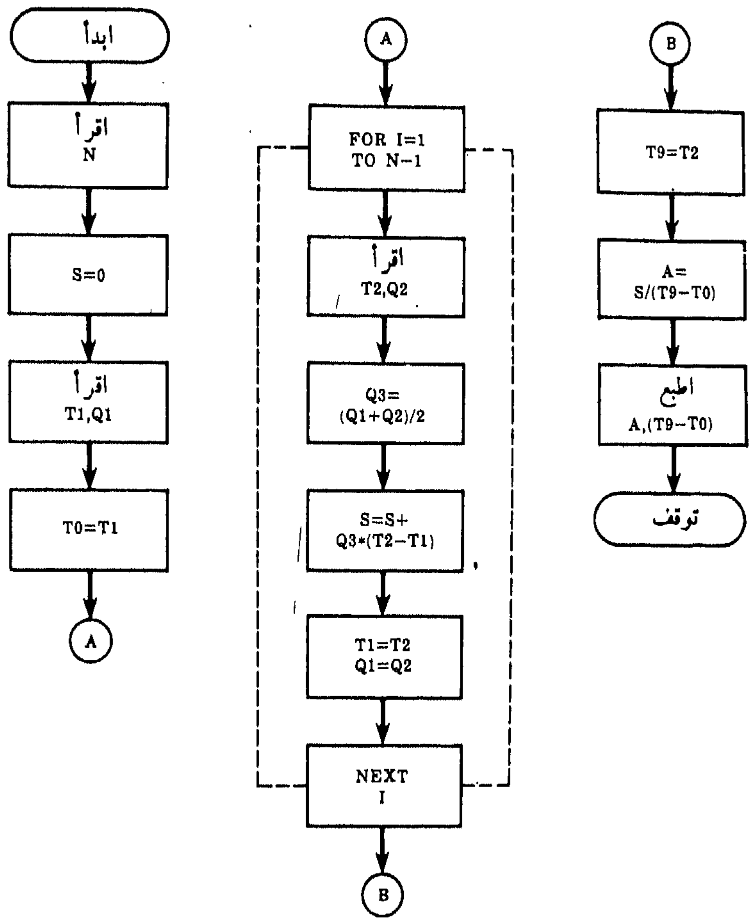
٩ – توقف .

يبين شكل ٤ – ٧ خريطة سير عمليات مناظرة للمخطط التمهيدى الأعلى . لاحظ أن الحلقة التكرارية FOR-TO....NEXT تم حصرها بداخل مستطيل من الخطوط المتقطعة .

برنامج البيسك The Basic Program

يبين شكل ٤ – ٨ برنامج بيسك كاملا يناظر المخطط التمهيدى الممثل عاليه ، نرى أن المجموع التراكمى S يتكون بواسطة حلقة FOR-TO....NEXT التى تتكون من الجمل 100 إلى 160. لاحظ أن البرنامج لا يتطلب تخزين القائمة الكاملة من بيانات الإدخال من أجل حساب هذا المجموع التراكمى (وسوف نرى طريقة مناسبة لتخزين قائمة كاملة ، إذا وجب علينا تنفيذ ذلك فى الفصل التالى) .

يبين الجزء الأدنى من شكل ٤ – ٨ قيم المدخلات والنتائج المحسوبة والتى تناظر البيانات فى الجدول ٤ – ١ . نرى أن المتوسط الشامل لنوعية الهواء هى 41.5 لفترة 24 ساعة المعطاة . (لاحظ أن الإجابة لن تكون مضبوطة لأكثر من 3 أرقام معنوية ، حيث أن هذه هى دقة البيانات المدخلة) وسنجد أن المعلومات المدخلة موضوع تحتها خط .



شكل ٤ – ٧

٤ – ٧ الحلقات التكرارية المتداخلة NESTED LOOPS

يمكن تسكين حلقة تكرارية بداخل حلقة تكرارية أخرى (أى تداخل) . إذا رغبنا ذلك . فى الحقيقة ، يمكن وجود عدة مستويات للتداخل وتطبق قواعد كتابة حلقة تكرارية مفردة أيضاً على الحلقات التكرارية المتداخلة . وبالأضافة لذلك يجب ملاحظة القيود التالية :

١ – يجب أن تبدأ كل حلقة متداخلة بجملة FOR-TO الخاصة بها وأن تنتهى بجملة NEXT الخاصة بها .

٢ – الحلقة التكرارية الخارجية والداخلية (المتداخلة) لا يمكن أن يكون لها نفس المتغير الجارى .

٣ – كل حلقة داخلية (متداخلة) يجب أن تسكن كاملا بداخل الحلقة التكرارية الخارجية (أى لا يمكن للحلقات التكرارية أن تتشابك) .

٤ – يمكن تحويل التحكم من حلقة داخلية (متداخلة) إلى جملة فى الحلقة التكرارية الخارجية أو إلى جملة خارج نطاق الحلقات المتداخلة بأكلها . ومع ذلك ، لا يمكن تحويل التحكم إلى جملة داخل الحلقات المتداخلة من نقطة خارج مدى الحلقات المتداخلة .

```
10 REM AVERAGING OF AIR POLLUTION DATA
20 PRINT "N=";
30 INPUT N
40 LET S=0
50 PRINT
60 PRINT "  T   Q"
70 PRINT
80 INPUT T1,Q1
90 LET T0=T1
100 FOR I=1 TO N-1
110     INPUT T2,Q2
120     LET Q3=(Q1+Q2)/2
130     LET S=S+Q3*(T2-T1)
140     LET T1=T2
150     LET Q1=Q2
160 NEXT I
170 LET T9=T2
180 LET A=S/(T9-T0)
190 PRINT
200 PRINT "AVERAGE AIR QUALITY=";A;"TIME INTERVAL=";T9-T0;"HOURS"
210 END

>RUN

EX4.16          21:58           14-MAR

N= ?20

  T   Q

?0,  12.2
?2,  12.5
?4,  11.9
?6,  13.5
?7,  22.4
?8,  31.4
?9,  57.7
?10, 84.4
?11, 68.0
?12, 51.6
?13, 46.6
?14, 43.1
?15, 39.2
?16, 44.7
?17, 62.9
?18, 88.0
?19, 71.4
?20, 57.0
?22, 43.5
?24, 28.7

AVERAGE AIR QUALITY= 41.5354 TIME INTERVAL= 24 HOURS

TIME:  0.63 SECS.
```

شكل ٤ – ٨

مثال ٤ – ١٧

يبين الشكل التالى هيكلا لبرنامج يحتوى على حلقة تكرارية متداخلة .

```
100 FOR I=0 TO N STEP 2
        ...
120     FOR J=I TO N
            ...
160     NEXT J
        ...
200 NEXT I
```

لاحظ أن الحلقة الداخلية (الجمل 120 إلى 160) موجودة بالكامل داخل الحلقة الخارجية (الجمل 100 إلى 200) . كل حلقة تكرارية تبدأ وتنتهي بجملتي FOR-TO و NEXT الخاصة بها . وكل حلقة تكرارية لها المتغير الجاري الخاص بها . ومع ذلك ، لاحظ أن المتغير الجاري للحلقة الخارجية (I) قد استخدم كقيمة أولية للمتغير الجاري للحلقة الداخلية (J) وهذا مسموح به مادامت قيمة I لا تتغير بداخل الحلقة الداخلية .

تعطي الحلقات التكرارية وسائل مناسبة للقيام بعمل تكرار مجموعة من الحسابات . وإليك المثال التالي :

مثال ٤ – ١٨ توليد أرقام فيبوناسي (Fibonacci) والبحث عن الأرقام الأولية :

المعروف أن أرقام فيبوناسي (Fibonacci) أعضاء لتسلسل شيق حيث كل رقم يساوي مجموع الرقمين السابقين . بمعنى آخر .

$$F_i = F_{i-1} + F_{i-2}$$

حيث تشير F_i للرقم الذي يحمل الترتيب i . رقما فيبوناسي (Fibonacci) الأوليان تم تعريفهما بالوحدة ١ أي

$$F_1 = ١$$

$$F_2 = ١$$

وعلى ذلك

$$F_3 = F_2 + F_1 = 1 + 1 = 2$$
$$F_4 = F_3 + F_2 = 2 + 1 = 3$$
$$F_5 = F_4 + F_3 = 3 + 2 = 5$$

وهكذا .

كل أعداد فيبوناسي كميات صحيحة موجبة ، وبعض منها سوف تكون أولية . والرقم الأولي الموجب هو رقم صحيح يقبل القسمة ، بدون باق ، على نفسه وعلى الواحد الصحيح فقط . فمثلا 5 رقم أولي لأن الكميات التي يقبل القسمة عليها بدون باق هي 5 و 1 . وبمعنى آخر 8 ليست رقماً أولياً لأن 8 تقبل القسمة على 2 و 4 بجانب 1 و 8 .

طريقة إجراء الحسابات Computational Procedure

من السهل حساب أول N رقم من أرقام فيبوناسي باستخدام الصيغة الرياضية أعلاه . ومع ذلك ، فإن الإجراء الذي يحدد ما إذا كان الرقم أولياً أو غير أولي يتطلب بعض الشرح .

ولنفرض أننا نريد تحديد أن رقماً معيناً قيمته أكبر من 2 يمكن قسمته بالتساوي على رقم صحيح أصغر . ولنسم الرقم الصحيح F والمقسوم عليه J وطريقة حساب خارج القسمة Q هي :

$$Q = F/J$$

وبعد ذلك نحسب خارج قسمة مبتوراً Q1 كما يلي :

$$Q1 = INT\ (Q)$$

وتشير الحروف INT لدالة بيسك المكتبية التي تحدد أكبر رقم صحيح لا يتعدى قيمة Q (وسوف نقول الكثير عن دوال بيسك المكتبية في الفصل التالي) . وبذلك إذا كانت Q لها القيمة 5.3 ، فإن Q1 سيكون لها القيمة 5 .

وإذا كانت Q1 و Q لهما نفس القيمة ستكون F مقسومة بالتساوي على J. وعلاوة على ذلك ، إذا قسمت F بالتساوي على أي قيمة للمتغير J من J = 2 إلى $J = INT\sqrt{F}$ ، فإن F لا يمكن أن تكون أولية . حيث F ستكون رقماً أولياً فقط إذا كانت Q1 و Q لا تساوي أي قيمة من $J = 2, 3, \ldots INT\sqrt{F}$

المخطط التمهيدى للبرنامج The Program Outline

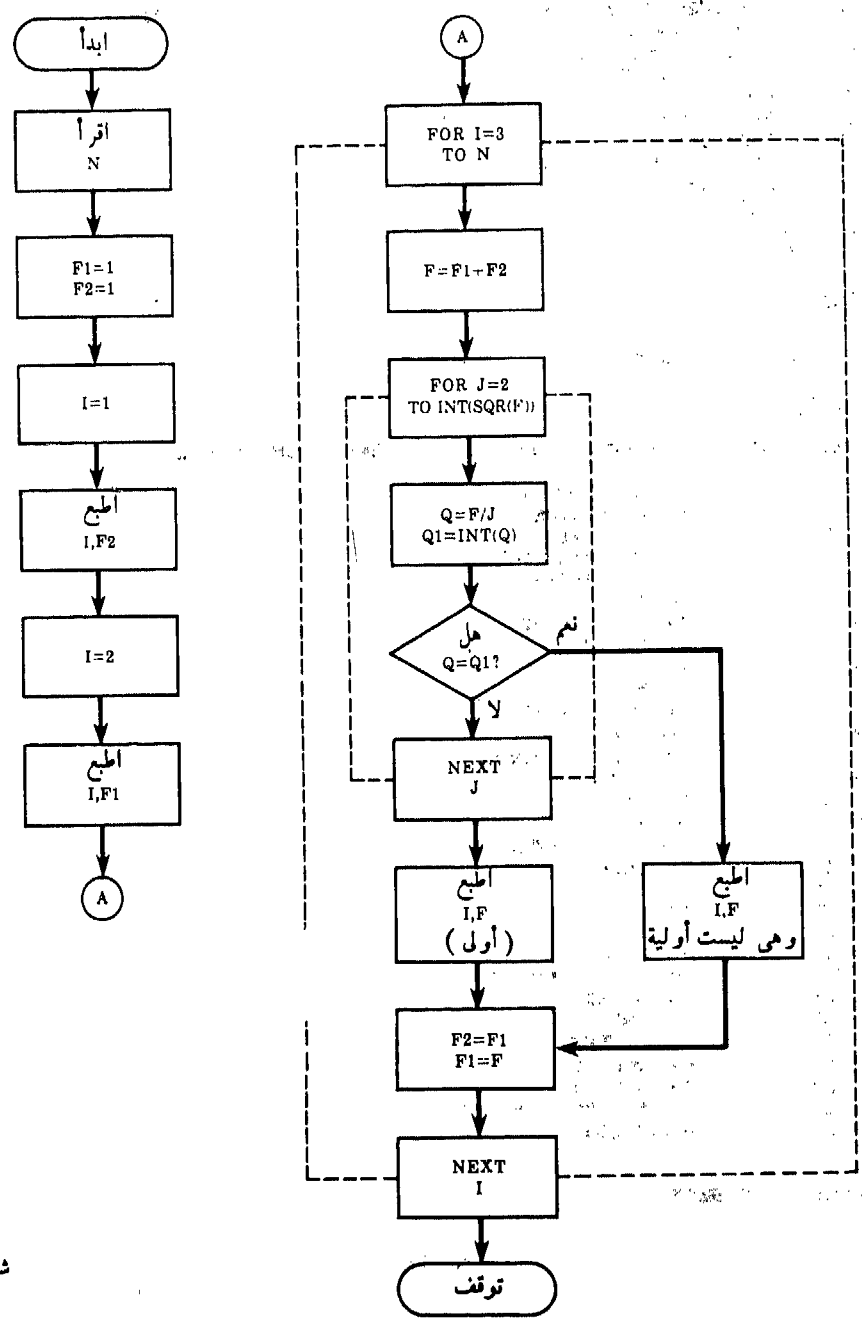
دعنا نشير إلى F كرقم فيبوناسى (أى F_i) و F1 كرقم فيبوناسى السابق (F_{i-1}) و F2 كرقم ثان سابق (F_{i-2}) . يمكننا الآن كتابة مخطط تمهيدى للبرنامج الخاص بنا كما يلى :

١ – اقرأ N

٢ – اجعل F_1 و F_2 مساوياً 1

٣ – اطبع F1 و F2 وتعرف على أن كلا منها رقم أولى .

٤ – كرر نفس الحسابات للقيم : I = 3, 4, ... N



شكل ٤ – ٩

```
10 REM GENERATION OF FIBONACCI NUMBERS AND SEARCH FOR PRIMES
20 PRINT "N=";
30 INPUT N
40 PRINT
50 PRINT "GENERATION OF FIBONACCI NUMBERS AND SEARCH FOR PRIMES"
60 PRINT
70 LET F1=1
80 LET F2=1
90 PRINT "I=";1,"F=";1;" (PRIME)"
100 PRINT "I=";2,"F=";1;" (PRIME)"
110 FOR I=3 TO N     '(GENERATE FIBONACCI NUMBERS)
120     LET F=F1+F2
130     FOR J=2 TO INT(SQR(F))     '(TEST FOR A PRIME NUMBER)
140        LET Q=F/J
150        LET Q1=INT(Q)
160        IF Q=Q1 THEN 200
170     NEXT J
180     PRINT "I=";I,"F=";F;" (PRIME)"
190     GOTO 210
200     PRINT "I=";I,"F=";F
210     LET F2=F1
220     LET F1=F
230 NEXT I
240 END

>RUN

EX4.18          09:26          15-MAR

N= ?30

GENERATION OF FIBONACCI NUMBERS AND SEARCH FOR PRIMES

I= 1           F= 1  (PRIME)
I= 2           F= 1  (PRIME)
I= 3           F= 2  (PRIME)
I= 4           F= 3  (PRIME)
I= 5           F= 5  (PRIME)
I= 6           F= 8
I= 7           F= 13  (PRIME)
I= 8           F= 21
I= 9           F= 34
I= 10          F= 55
I= 11          F= 89  (PRIME)
I= 12          F= 144
I= 13          F= 233  (PRIME)
I= 14          F= 377
I= 15          F= 610
I= 16          F= 987
I= 17          F= 1597  (PRIME)
I= 18          F= 2584
I= 19          F= 4181
I= 20          F= 6765
I= 21          F= 10946
I= 22          F= 17711
I= 23          F= 28657  (PRIME)
I= 24          F= 46368
I= 25          F= 75025
I= 26          F= 121393
I= 27          F= 196418
I= 28          F= 317811
I= 29          F= 514229  (PRIME)
I= 30          F= 832040

TIME:  0.35 SECS.
```

شكل ٤ — ١٠

(أ) احسب قيمة للمتغير F باستخدام الصيغة الرياضية .

$$F = F1 + F2$$

(ب) افعل ذلك لقيم $J = 2, 3, \ldots\ldots, R$ حيث R أكبر رقم صحيح لا يتعدى $\sqrt{F}$ (اختبار الرقم الأولى)

(i) احسب قيما للمتغيرات Q و Q1 واختبر ما إذا كانا متساويان .

(ii) إذا تساوت Q و Q1 لأى قيمة من J فإن F لا يمكن أن تكون رقماً أولياً . حيث تطبع I و F ، واصل مباشرة الخطوة ٤ (جـ) التالية .

(iii) أما إذا كانت Q لا تساوى Q1 لكل قيم J فإن F يجب أن تكون أولية . حيث تطبع I و F وتعرف بأنها أولية ، واصل لخطوة ٤ (جـ) التالية .

(جـ) عدل قيمتى F1 و F2 (أى حدد القيمة F2 بالقيمة الحالية F1 ثم حدد قيمة F1 بالقيمة الحالية F) وذلك للتجهيز لحساب رقم فيبوناسى جديد (أى قيمة جديدة للمتغير F) .

٥ – توقف

يبين شكل ٤ – ٩ خريطة سير عمليات مناظرة .

برنامج البيسك The BASIC Program

يبين شكل ٤ – ١٠ برنامج بيسك كاملا مناظراً للمخطط التمهيدى السابق . لاحظ أن البرنامج يحتوى على تداخل فى الحلقات التكرارية . الغرض من الحلقة الداخلية (الجمل 130 إلى 170) هو تحديد ما إذا كان رقم فيبوناسى أولياً أم لا ، بينما تسبب الحلقة الخارجية (الجمل 110 إلى 230) توليد أرقام فيبوناسى المتتالية . لاحظ استخدام الدالة المكتبية SQR (فى السطر 130)، وقد استخدمت لحصول على الجذر التربيعى للمتغير F (انظر القسم ٥ – ١) . ولاحظ أيضاً ، التحول المشروط للتحكم إلى خارج الحلقة الداخلية إذا كانت $Q = Q1$ (الجملة 160) .

يبين الجزء الأسفل من شكل ٤ – ١٠ المخرجات التى تم توليدها عند تنفيذ البرنامج للقيمة $N = 30$. نلاحظ أن 11 رقماً من الثلاثين رقماً الأولى لفيبوناسى أرقاماً أولية . (لاحظ أن اجابات المستخدم موضوع تحتها خط) .

اسئلة للمراجعة

Review Questions

٤ – ١ هل اتخاذ قرار منطقى من الصفات الهامة للحاسبات الرقمية ؟ اشرح الأسباب لإجابتك .

٤ – ٢ ما هو المقصود بتحويل التحكم بداخل برنامج بيسك ؟

٤ – ٣ ما هى عملية التفرع المشروطة ؟ وكيف تختلف هذه العملية عن عملية التفرع غير المشروطة ؟

٤ – ٤ ما هى عملية التكرار ؟ وما هو الغرض من مثل هذه العملية ؟

٤ – ٥ اذكر أسماء المعاملات الست الرابطة المستخدمة فى البيسك ؟ وما هو الغرض منها ؟

٤ – ٦ ما هو الغرض من جملة IF-THEN ؟

٤ – ٧ لخص قواعد كتابة جملة TF-THEN واشرح ماذا يحدث عند تنفيذ هذه الجملة .

٤ – ٨ كيف تختلف جملة IF-THEN عن جملة GO TO ؟ وهل يمكن استخدام هاتين الجملتين سوياً للقيام بتنفيذ عملية منطقية مشتركة ؟

٤ – ٩ ما هو المقصود من إجراء تكرارى ؟

٤ – ١٠ ما هو الغرض من جملة ON-GO TO ؟

٤ – ١١ كيف تختلف جملة ON-GO TO عن جملة IF-THEN ؟

٤ – ١٢ لخص قواعد كتابة جملة ON-GO TO واشرح ماذا يحدث عند تنفيذ هذه الجملة .

٤ – ١٣ ماذا يحدث بالضبط عند بتر الرقم ؟

٤ – ١٤ ما هو الغرض من جملة STOP ؟ وكيف تختلف عن جملة END ؟

٤ – ١٥ ما هو الغرض من جملة FOR-TO ؟ وما هو الغرض من المتغير الجارى وعبارة STEP ؟

٤ – ١٦ هل يمكن أن يأخذ المتغير الجارى قيمة عشرية (كسر) أو قيمة سالبة ؟

٤ – ١٧ هل يمكن إنقاص قيمة المتغير الجارى عند تنفيذ الحلقة التكرارية مرات متتالية ؟

٤ – ١٨ ما هو الغرض من جملة NEXT ؟ ما هو المطلوب من المتغير الجارى الذى يظهر فى هذه الجملة ؟

٤ – ١٩ اذكر طريقتين مختلفتين يمكن أن تكون بهما الحلقة التكرارية فى البيسك واذكر نوع الموقف الذى يتناسب مع كل نوع من أنواع الحلقات التكرارية ؟

٤ – ٢٠ هل يمكن للمتغير الجارى أن يظهر فى جملة من الجمل التى تحتويها الحلقة التكرارية بداخل FOR-TO.....NEXT ؟ وهل يمكن تغيير قيمة المتغير الجارى فى مثل هذه الجملة ؟

٤ – ٢١ تحت أى ظروف سوف تنفذ حلقة FOR-TO......NEXT مرة واحدة ؟ أو لا تنفذ على الإطلاق ؟

٤ – ٢٢ ما هي القيود التى تطبق على تحويل التحكم من أو إلى حلقة FOR-TO.....NEXT ؟

٤ – ٢٣ لماذا تظهر الجمل فى حلقة FOR-TO.....NEXT دائماً مزحزحة إلى الداخل ؟ وهل هذه الزحزحة مهمة ؟

٤ – ٢٤ ما هو المقصود من الحلقات التكرارية المتداخلة ؟

٤ – ٢٥ لخص القواعد التى تطبق لتداخل الحلقات التكرارية FOR-TO......NEXT قارن ذلك بالقواعد التى تطبق على حلقة تكرارية مفردة .

مسائل محلولة

Solved Problems

٤ – ٢٦ كل شرط من الشروط التالية يتضمن استخدام المعاملات الرابطة . تعرف على أى من هذه الشروط . مكتوب بطريقة غير صحيحة إن وجد .

الشرط	الخطأ
X="DATE"	لا يمكن مقارنة متغير رقمى مع متغير حرفى .
K↑2>=100	صحيح
N$<>A+B	لا يمكن مقارنة متغير حرفى بكمية عددية .
P$="123456"	صحيح
T$=R$+S$	لا يسمح بالصيغ الرياضية التى تحتوى حروفاً أو متغيرات حرفية .

٤ – ٢٧ مبين فيما يلى عدة جمل IF-THEN حدد ، إذا كانت أى منها قد كتبت بطريقة غير صحيحة .

الجملة	الخطأ
20 IF K↑2>=100 THEN 50	صحيحة
20 IF (K↑2>=100) THEN 50	صحيحة
20 IF K↑2>=100 GO TO 50	ليست كل نسخ بيسك (BASIC) تسمح باستخدام GO TO بدلا من THEN
65 IF X+Y<>Z THEN M	يجب أن يكون رقم الجملة التى يتحول إليها التحكم رقماً صحيحاً موجباً وليس متغيراً .
100 IF G$="MAY 13" THEN 45	صحيحة
35 GO TO 150 IF J=3	بناء هيكل غير صحيح لغوياً (يجب أن تبدأ الجملة بالكلمة IF) .
50 IF X1<=50 THEN X=X+5	بناء هيكل غير صحيح لغوياً (يجب أن تتبع THEN رقم جملة فقط) .

٤ – ٢٨ مبين فيما يلى البناء الهيكلى لعدة جمل IF-THEN....GO TO والتى تكون حلقات تكرارية . حدد ، إذا كانت أى منها قد كتبت بطريقة غير صحيحة .

(أ)

```
20...
...
60 IF N>N1 THEN 110
...
85 LET N=N+1
90 GO TO 20
...
110...
```

صحيحة ________________

(ب)

```
35 LET P=0

50...
...
75 IF P<=0 THEN 125
80 GO TO 50
...
125...
```

____

يتحول التحكم دائماً إلى الجملة رقم 125 ، حيث تم تحديد قيمة مبدئية للمتغير P بصفر ولن تتغير قيمتها فيما بعد .

(جـ)

```
15...
...
45 INPUT X
50 IF X<50 THEN 90
...
85 STOP
90...
...
110 GO TO 15
```

________________ صحيحة

(د)

```
110 C=5
...
130...
...
160 IF C<=0 THEN 200
170 C=C+5
180 GO TO 130
...
200...
```

سوف تستمر الحلقة التكرارية إلى ما لا نهاية حيث لن تكون قيمة C أقل أو تساوى صفر إطلاقاً .

٤ – ٢٩ مبين فيما يلى عدة جمل ON-GO TO حدد ، إذا كانت أى منها قد كتبت بطريقة غير صحيحة .

الجملة	الخطأ
15 ON X3 GO TO 25,15,25,40	صحيحة
100 ON 2*(C1+C2)/N GO TO K1,K2,K3	يجب أن تكون أرقام الجمل التى يتحول إليها التحكم أرقاماً صحيحة موجبة . وليست متغيرات
55 ON J+K THEN 120,90,150	لا تسمح كل نسخ بيسك باستخدام THEN بدلا من GO TO
80 ON T GO TO 25	يجب أن تعطى أرقام جملتين مختلفتين على الأقل .
20 ON N$ GO TO 50,70,50,90	لا يمكن لمتغير حرفى أن يظهر فى جملة ON-GO TO .
60 ON P(I) GO TO 10,120	صحيحة (لاحظ أن P(I) متغير ذو دليل) وسوف نناقش المتغيرات ذات الأدلة فى الفصل الخامس .

٤ – ٣٠ مبين فيما يلى عدة جمل FOR-TO حدد ، إذا كانت أى منها قد كتبت بطريقة غير صحيحة .

الجملة	الخطأ
20 FOR N=J TO 3*(K+1) STEP J1	صحيحة
50 FOR N=0 TO −100 STEP −5	صحيحة
75 FOR X(I)=1 TO 100	لا يمكن استخدام متغير ذى دليل كمتغير جار .
35 FOR K=K1 TO 100 STEP K1	صحيحة
65 FOR N$=1 TO 19 STEP 2	لا يمكن استخدام متغير حرفى كمتغير جار .
100 FOR I=J TO K STEP I	لا يمكن أن يظهر المتغير الجارى فى عبارة STEP

٤ – ٣١ مبين فيما يلى عدة هياكل لحلقات التكرارية FOR-TO......NEXT حدد إذا كانت أى منها ، مكتوبة بطريقة غير صحيحة

(أ)

```
20 FOR I=1 TO 100 STEP J

80 NEXT J
```

المتغير الجارى الموجود فى جملة NEXT وهو (J) . ليس هو نفس المتغير الجارى فى جملة FOR-TO وهو (I) .

(ب)

```
50 FOR N=N1 TO N2
...
75...
...
90 NEXT N
...
120 IF N=10 THEN 75
```

لا يمكن أن يتحول التحكم إلى داخل الحلقة التكرارية .

(ج)

```
100 FOR K=3 TO -3 STEP -1
...
130 PRINT X↑K
...
150 NEXT K
```

صحيحة ، بفرض أنه تم تحديد قيمة للمتغير X . (لاحظ ظهور المتغير الجارى بداخل جملة PRINT ، ولكن قيمته لم تتغير) .

(د)

```
 25 FOR X=0 TO 1 STEP 0.05
        ...
 50     FOR Y=0 TO 10 STEP 0.1
            ...
 75     NEXT X
        ...
100 NEXT Y
```

تشابك الحلقة التكرارية الداخلية والحلقة التكرارية الخارجية .

(هـ)

```
100 FOR I=1 TO M
        ...
125     FOR J=1 TO N
            ...
135         FOR K=1 TO M+N
                ...
160         NEXT K
            ...
180     NEXT J
        ...
200     FOR J=1 TO N
            ...
225     NEXT J
        ...
235     FOR K=1 TO M+N
            ...
240     NEXT K
        ...
250 NEXT I
```

صحيحة

مسائل تكميلية

Supplementary Problems

٤ – ٣٢ يتضمن كل شرط مما يلي استخدام المعاملات الرابطة . حدد إذا كانت أي منها قد كتبت بطريقة غير صحيحة .

(ا) J>=J1+J2 (د) X<.01

(ب) C=C+1 (هـ) P2<>T$

(جـ) N$="END" (و) A/B<=C/D

٤ – ٣٣ ما هو المطلوب من أجل أن يتحقق الشرط :

P$<Q$

في كل من السلاسل الحرفية الممثلة بالمتغيرات P$ و Q$ ؟

٤ – ٣٤ مبين فيما يلي عدة جمل IF-THEN حدد إذا كانت أي منها قد كتبت بطريقة غير صحيحة .

(ا) 30 IF K<>K1 THEN 10

(ب) 50 IF J<100 THEN J=J+1

(جـ) 120 IF X>=Y+Z GO TO 200

(د) 75 IF M$="DATE" THEN 50

(هـ) 150 IF P=P1 THEN K

(و) 100 IF B↑2<>(4*A*C) THEN 150

(ز) 45 IF X↑2<0 THEN 20

٤ – ٣٥ اكتب جملة بيسك أو أكثر لتوافق كل موقف من المواقف الموصوفة التالية :

(ا) حول التحكم للجملة رقم 50 إذا كانت قيمة K أقل من 15 وإلا نفذ الجملة التالية .

(ب) حول التحكم للجملة رقم 70 إذا كانت قيمة N$ مساوية للسلسلة "OPTION A" وإلا حول التحكم إلى الجملة رقم 150 .

(جـ) حول التحكم للجملة رقم 200 إذا كانت قيمة X أقل من أو تساوى 100 ؛ وإلا أضف مقدار الوحدة على المتغير J ، واقرأ قيمة جديدة للمتغير X ، ثم ارجع مرة ثانية للجملة 60 .

(د) حول التحكم للجملة رقم 150 إذا كانت قيمة J مساوية للصفر ، وإلا أضف قيمة J على القيمة الموجودة في S ، ثم ارجع مرة ثانية للجملة رقم 20 .

٤ – ٣٦ مبين فيما يلي البناء الهيكلي لعدة جمل IF-THEN.....GO TO والتي تكون حلقات تكرارية . حدد إذا كانت أي منها قد كتبت بطريقة غير صحيحة .

(أ)
```
20...
...
120 IF K$="END" THEN 200
130 PRINT A,B,C
140 GO TO 20
...
200 STOP
```

(ب)
```
10 INPUT N
20 LET T=0
30 LET J=1
40 INPUT T1
50 LET T=T+T1
60 IF J=N THEN 150
70 LET J=J+1
80 GO TO 30
...
150 PRINT J,T
160 END
```

(ج)
```
10 LET X=100
...
50...
...
100 IF X>=100 THEN 200
110 LET X=X-5
120 GO TO 50
...
200...
```

(د)
```
10 LET X=100
...
50...
...
100 IF X>100 THEN 200
110 LET X=X-5
120 GO TO 50
...
200...
```

(ﻫ)
```
10 LET X=100
...
50...
...
100 IF X=0 THEN 200
110 LET X=X-5
120 GO TO 50
...
200...
```

٤ – ٣٧ مبين فيما يلي عدة جمل ON-GO TO . حدد إذا كانت أي منها قد كتبت بطريقة غير صحيحة .

(أ) `50 ON N$ GO TO 10,70,120`
(ب) `100 ON A GO TO 50,20,50,200,120`
(ج) `75 ON X1 GO TO 100,25,75,150`
(د) `100 ON N GO TO N1,N2,N3,N4,N5`
(ﻫ) `150 ON ((A+B)/C)↑2 GO TO 170,190,225`
(و) `45 ON K-3 THEN 65,90`
(ز) `80 GO TO 50,120,150 ON (X+Y)`

٤ – ٣٨ ادرس الجملة ON-GO TO التالية (وهي صحيحة لغوياً) :

```
75 ON J-K GO TO 100,50,20,150
```

ماذا يحدث عند تنفيذ هذه الجملة إذا كانت :

(أ) J=2 and K=3?
(ب) J=4.5 and K=0.75?
(ج) J=1 and K=-1?
(د) J=5 and K=0?
(ﻫ) J=-1 and K=-2.8?

٤ – ٣٩ مبين فيما يلي عدة جمل FOR-TO حدد إذا كانت أي منها قد كتبت بطريقة غير صحيحة .

(أ) `100 FOR C=.1*A TO .25*(A+B) STEP P/2`
(ب) `65 FOR X↑2=0 TO 100 STEP 20`
(ج) `80 FOR K$=P$ TO Q$`
(د) `125 FOR X=V(1) TO V(2) STEP V(3)`
(ﻫ) `30 FOR J1=25 TO -25 STEP -5`
(و) `45 FOR A=12 TO 0 STEP 0.1`

٤ – ٤٠ اكتب زوجاً مناسباً من جمل FOR-TO و NEXT (أى اكتب هيكل حلقة تكرارية FOR-TO......NEXT لكل من المواقف الموصوفة التالية :

(ا) حلقة تكرارية يجب تكرارها 200 مرة .

(ب) حلقة تكرارية يجب تكرارها 200 مرة . ماعدا أن التحكم سوف يتحول خارج الحلقة إلى جملة 175 إذا أصبحت قيمة X أقل من 0.001

(ج) حلقة تكرارية تكرر عدداً مناسباً من المرات للمتغير الجارى الذى يتزايد من 1 إلى 73 ، بفرض أن المتغير الجارى يزاد فى كل مرة بمقدار 3 وحدات عند تنفيذ الحلقة التكرارية .

(د) حلقة تكرارية تكرر عدداً مناسباً من المرات للمتغير الجارى الذى يتزايد من 0.5 إلى قيمة معطاة بالصيغة الرياضية 10 — 3 ↑ A. فى كل مرة تنفذ فيها الحلقة التكرارية تزاد قيمة المتغير الجارى بالقيمة المعطاة فى الصيغة الرياضية A + B .

٤ – ٤١ مبين فيما يلى هياكل عدة حلقات تكرارية FOR-TO......NEXT ، حدد إذا كانت أى منها قد كتبت بطريقة غير صحيحة :

(ا)
```
10 FOR K=K1 TO K2
      ...
50      K=K+1
      ...
90 NEXT K
```

(ب)
```
10 FOR C1=0 TO 50 STEP 5
      ...
35      FOR C2=0 TO C1
          ...
65      NEXT C2
      ...
100 NEXT C1
```

(ج)
```
10 FOR J=1 TO N
      ...
40      FOR K=1 TO M
          ...
70      NEXT J
      ...
120 NEXT K
```

(د)
```
100 FOR X=0 TO 1 STEP .02
      ...
150      FOR X=0 TO 5 STEP .05
      ...
200 NEXT X
```

(هـ)
```
50 FOR P=1 TO 50
      ...
80      FOR Q=2 TO 100 STEP 2
          ...
100          IF T>=T1 TH   160
          ...
120      NEXT Q
      ...
160 PRINT "T=";T
      ...
200 NEXT P
```

(و)
```
75 FOR X=A TO (A+B) STEP C
      ...
125 NEXT C
```

مسائل للبرمجة

Programming Problems

٤ – ٤٢ يمكن إعادة تنظيم المعادلة

$$x^5 + 3x^2 - 10 = 0$$

الممثلة فى المثال ٤ – ٥ بالصورة التالية :

$$x = \sqrt{(10 - x^5)/3}$$

أعد كتابة برنامج البيسك المعروض فى مثال ٤ – ٥ حتى يتسنى له استخدام الصيغة السابقة للمعادلة . نفذ البرنامج مع طباعة قيم x المحسوبة أثناء كل تكرار . قارن النتائج المحسوبة مع النتائج المعروضة فى مثال ٤ – ٥ .

٤ – ٤٣ أعد كتابة البرنامج المعروض فى مثال ٥ – ٤ حتى يتسنى له استخدام الحلقة التكرارية FOR-TO....NEXT فضلا عن الحلقة التكرارية IF-THEN......GO TO .

٤ – ٤٤ أعد كتابة البرنامج المعروض فى مثال ٤ – ٩ وذلك لاستبدال جمل ON-GO TO بجملة IF-THEN. ما هى الفائدة التى تم الحصول عليها باستخدام جمل ON-GO TO ؟

٤ – ٤٥ أعد كتابة البرنامج المعروض فى مثال ٤ – ١٦ حتى يتسنى له استخدام الحلقة التكرارية IF-THEN.......GO TO فضلا عن الحلقة التكرارية FOR-TO.....NEXT

٤ – ٤٦ اكتب برنامجاً من النوع التخاطبى الذى يقرأ رقماً صحيحاً موجباً ويحدد :

(ا) ما إذا كان الرقم الصحيح رقماً أولياً .

(ب) ما إذا كان الرقم الصحيح ضمن أرقام فيبوناسى (Fibonacci).

اكتب البرنامج بطريقة تمكنه من تكرار التنفيذ (حلقة تكرارية) ، حتى تكون الكمية المدخلة مساوية صفراً .

٤ – ٤٧ نعرض فيما يلى عدة تمارين بسيطة للبرمجة . جهز مخططاً تمهيدياً مفصلا وخريطة سير عمليات مناظرة وبرنامج بينسك كاملا لكل من هذه التمارين . نفذ البرنامج باستخدام البيانات المعطاة .

(ا) احسب المتوسط الحسابى لقائمة من عدد n من الأرقام . اختبر البرنامج الخاص بك بمجموعة البيانات التالية $(n = 10)$

27.5	87.0
13.4	39.9
53.8	47.7
29.2	8.1
74.5	63.2

(ب) احسب المتوسط الحسابى المرجح لقائمة تتكون من عدد n من الأرقام باستخدام الصيغة الرياضية :

$$x_{avg} = f_1x_1 + f_2x_2 + \cdots + f_nx_n$$

حيث قيم f هى معاملات مرجحة كسرية أى أن $0 \le n_i \le 1$ و $f_1 + f_2 + \ldots f_n = 1$. اختبر البرنامج الخاص بك باستخدام البيانات المعطاة فى الجزء (أ) والمعاملات المرجحة التالية $(n = 10)$

$i = 1$	$f = 0.06$	$i = 6$	$f = 0.10$
2	0.08	7	0.12
3	0.08	8	0.12
4	0.10	9	0.12
5	0.10	10	0.12

(ﺟ) احسب حاصل الضرب التراكمى لقائمة n من الأرقام . اختبر البرنامج الخاص بك بالمجموعة التالية من البيانات $(n = 6)$:

6.2 و 12.3 و 5.0 و 18.8 و 7.1 و 12.8

(د) احسب المتوسط الهندسى لقائمة من الأرقام باستخدام الصيغة الرياضية :

$$x_{avg} = [x_1x_2x_3 \cdots x_n]^{1/n}$$

اختبر البرنامج الخاص بك باستخدام البيانات المعطاة فى الجزء (ﺟ) عاليه . قارن بين النتائج التى تم الحصول عليها من المتوسط الحسابى لنفس البيانات . أى من المتوسطات أكبر ؟

٤ – ٤٨ جهز مخططاً تمهيدياً مفصلا ، وخريطة سير عمليات مناظرة وبرنامج بيسك كاملا لكل من المسائل المعروضة التالية :

(ا) احسب مجموع أول 100 رقم صحيح فردى أى ($1 + 3 + 5 .... + 199$) اكتب البرنامج بطريقتين مختلفتين :

(i) اكتب مجموعة متتالية من الجمل والتى تجمع الرقم الصحيح التالى على المجموع فى كل مرة يعاد فيها التسلسل . دع القرار لتكرار التسلسل أو للإنتهاء متوقفاً على ناتج جملة IF-THEN

(ii) استخدم حلقة تكرارية FOR-TO...... NEXT

ما هى المزايا والعيوب ؟ صف خصائص كل طريقة .

(ب) احسب مجموع حاصل ضرب عدد N مضروباً فى رقم صحيح K . (فمثلا ، إذا كانت $K = 3$ و $N = 10$ فاحسب مجموع $1 \times 3 + 2 \times 3 + 3 \times .... 10 \times 3$) اجعل البرنامج عاماً تماماً وذلك بقراءة قيم N و K فى كل مرة ينفذ فيها البرنامج . اختبر البرنامج بحساب مجموع أول 1000 من مضاعفات الرقم الصحيح 3 .

(جـ) احسب قيمة !K ، حيث K تمثل رقماً صحيحاً تقرأ قيمته للحاسب كل مرة ينفذ فيها البرنامج . اختبر البرنامج بحساب قيمة !10 (لاحظ أن !K تعرف بـ $1 \times 2 \times 3 \times 4 ... \times K$)

(د) يمكن حساب جيب الزاوية x تقريباً وذلك بتجميع أول N حد من حدود المتوالية اللانهائية :

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots \quad \text{(تعطى قيم } x \text{ بالتقدير الدائرى)} \qquad (1)$$

اكتب برنامج بيسك يقرأ قيمة للمتغير x وبعد ذلك يحسب الجيب الخاص به . اكتب البرنامج بطريقتين مختلفتين .

(i) اجمع أول N من الحدود ، حيث تمثل N رقماً صحيحاً موجباً يقرأ فى الحاسب مع قيمة عددية للمتغير x .

(ii) استمر فى إضافة الحدود المتعاقبة فى المتوالية حتى تصبح قيمة الحد أصغر فى المقدار (من 10^{-5}) .

اختبر البرنامج للقيم $x = 1$ و $x = 2$ و $x = 3$ وفى كل حالة اكتب عدد الحدود المستخدمة بجانب الإجابة النهائية .

(هـ) نفرض أنك وضعت كية معينة من النقود A فى حساب ادخار عند بداية كل سنة لعدد n من السنوات . فإذا كان الحساب يربح بسعر فائدة مئوية i سنوياً ، وبذلك فإن كية النقود التى تم تراكمها بعد n من السنوات (F) معطاة بالصيغة :

$$F = A\,[\,(1+i/100) + (1+i/100)^2 + (1+i/100)^3 + \cdots + (1+i/100)^n]$$

اكتب برنامج بيسك من النوع التخاطبى ليحدد ما يلى :

(i) كم تكون كية النقود التى تتراكم بعد 30 سنة إذا كان المبلغ المودع 1000$ عند بداية كل سنة ونسبة الربح هى 6% سنوياً (حساب مركب سنويا) ؟

(ii) ما هو المبلغ الذى يجب أن يودع عند بداية كل سنة حتى يتراكم إلى 100,000$ بعد 30 سنة (مرة أخرى نفترض أن نسبة الربح هى 6% سنوياً ، (مركبة سنوياً) ؟

فى كل حالة ، ابدأ أولا بتحديد المبلغ غير المعروف . ثم بعد ذلك أنشئ جدولا مبيناً إجمالى المبلغ الذى تم تراكمه عند نهاية كل سنة

(و) عدل البرنامج السابق ليتلاءم مع حساب الأرباح المركبة ربع سنوية وليس سنوياً . قارن النتائج المحسوبة بالنتائج التى تم الحصول عليها فى الجزء (هـ) . تلميح الصيغة الرياضية الصحيحة هى :

$$F = A\,[\,(1+i/100m)^{m} + (1+i/100m)^{2m} + (1+i/100m)^{3m} + \cdots + (1+i/100m)^{nm}]$$

حيث تمثل m عدد فترات الأرباح لكل سنة .

(ز) حصل فصل من الطلبة على الدرجات التالية لإمتحانات السنة المأخوذة فى مقرر لغة البرمجة بيسك .

الاسم	درجات الاختبارات (نسبة مئوية)					
Adams	45	80	80	95	55	75
Brown	60	50	70	75	55	80
Davis	40	30	10	45	60	55
Fisher	0	5	5	0	10	5
Hamilton	90	85	100	95	90	90
Jones	95	90	80	95	85	80
Ludwig	35	50	55	65	45	70
Osborne	75	60	75	60	70	80
Prince	85	75	60	85	90	100
Richards	50	60	50	35	65	70
Smith	70	60	75	70	55	75
Thomas	10	25	35	20	30	10
Wolfe	25	40	65	75	85	95
Zorba	65	80	70	100	60	95

اكتب برنامج بيسك بصيغة تخاطبية يقبل اسم الطالب ودرجاته كدخلات ويحسب متوسط الدرجات لكل طالب . المطلوب أن يكون البرنامج عاماً بقدر الإمكان .

(ح) عدل البرنامج المكتوب فى جزء (ز) السابق ليسمح بأوزان غير متساوية لدرجات الاختبار الفردية . عموماً ، افرض أن كلا من الامتحانات الأربعة الأولى تساهم بمقدار 15% للدرجة النهائية ، وأن كلا من الامتحانين الآخرين تساهم بمقدار 20% .

(ط) أضف إلى البرنامج المكتوب فى الجزء (ح) السابق حتى يتسنى حساب متوسط شامل للفصل بالإضافة للمتوسطات الفردية للطلبة .

(ى) اكتب برنامج بيسك يسمح باستعمال النهاية الطرفية المركزية كآلة حاسبة مركزية . أخذاً فى الاعتبار العمليات الحسابية الشائعة فقط (الجمع والطرح والضرب والقسمة) .

(ك) اكتب برنامج بيسك لحساب جذور المعادلة التربيعية :

$$ax^2 + bx + c = 0$$

(انظر الأمثلة ٢ – ٢٦ و ٢ – ٣٠) اسمح بإمكانية جعل أحد الثوابت له القيمة صفر ، وأن الكمية $b^2 - 4ac$ تكون أقل من أو تساوى صفرا .

الفصـل ٥

بعض الخصائص الإضافية شائعة الاستعمال للغة البيسك
Some Additional Features of BASIC

يعرض هذا الفصل بعض الخصائص الإضافية شائعة الاستعمال للغة البيسك . وسنبدأ بمناقشة الدوال المكتبية المبنية داخلياً والتي تبسط بعض العمليات الحسابية الشائعة مثل حساب القيمة المطلقة لرقم ولوغاريتم عدد و . . . الخ . ثم بعد ذلك سوف ندرس استخدام القوائم والجداول التي تسمح لنا بتداول مجموعة من الكميات العددية أو الحرفية كما لو كانت متغيرات فردية . وأخيراً سوف نعرض جملتين إضافيتين لإدخال البيانات ، وبذلك ، تمدنا هاتان الجملتان بطريقة بديلة لاستخدام جملة INPUT .

٥ ـ ١ الدوال المكتبيـة LIBRARY FUNCTIONS

تمدنا الدوال المكتبية للغة البيسك (وتسمى أيضاً دوال عامة أو دوال أولية) بطريقة سريعة وسهلة لحساب عده دوال حسابية وللقيام بعمليات منطقية معينة . هذه الدوال المكتبية عبارة عن برامج صغيرة فرعية سبق كتابتها وتعتبر كجزء متكامل من اللغة . يمكن التوصل لأى دالة ببساطة بذكر إسمها ثم يلى الاسم أى معلومات يجب أن تعطى للدالة محصورة بين قوسين . (الكمية العددية أو سلسلة الحروف التى تصل للدالة بهذه الطريقة تسمى **خلاصة**) . وفور التوصل إلى الدالة فسوف تنفذ العملية الحسابية المطلوبة أوتوماتيكيا بدون الحاجة لكتابة برنامج مفصل لها .

مثال ٥ ـ ١

نفرض أن المطلوب حساب الجذر التربيعى للقيمة الممثلة بالمتغير X . فيمكننا كتابة :

```
50 LET Y=SQR(X)
```

وذلك يسبب تحديد قيمة Y بقيمة الجذر التربيعى للمتغير X. وتسمى الدالة التى تحسب الجذر التربيعى SQR ، والخلاصة فى هذا المثال هو المتغير X .

كان من الممكن بالطبع كتابة :

```
50 LET Y=X↑.5
```

وبذلك فاستخدام دالة الجذر التربيعى غير مطلوب . ولكنه مجرد شىء مفيد . (ويجب الإشارة مع ذلك ، إلى أن حساب الجذر التربيعى للرقم باستخدام دالة الجذر التربيعى تستغرق من وقت الحاسب كمية أقل من عملية الأس المناظرة) .

يمثل جدول ٥ ـ ١ عدة دوال مكتبية شائعة . وهناك قائمة أكثر مبينة فى الملحق ب ، وتتضمن أيضاً المداخل الموجودة فى جدول ٥ ـ ١ .

جدول ٥ — ١ دوال مكتبية شائعة الاستخدام

الدالة	التطبيق	الوصف
ABS	10 LET Y=ABS(X)	تحسب القيمة المطلقة للمتغير x ، $y = \|x\|$
ATN	10 LET Y=ATN(X)	تحسب مقابل الظل للمتغير x ، $y = \arctan(x)$
COS	10 LET Y=COS(X)	تحسب جيب تمام المتغير x ، $y = \cos(x)$ حيث x بالتقدير الدائرى
COT	10 LET Y=COT(X)	تحسب ظل تمام المتغير x ، $y = \cot(x)$ حيث x بالتقدير الدائرى .
EXP	10 LET Y=EXP(X)	ترفع e للقوة x ؛ $y = e^{x}$+
INT	10 LET Y=INT(X)	تحديد قيمة y بأكبر رقم صحيح حيث لا يتعدى قيمة x .
LOG	10 LET Y=LOG(X)	تحسب قيمة اللوغاريتم الطبيعى للمتغير x ، $y = \log_e x$ و $x > 0$
SGN	10 LET Y=SGN(X)	تحدد إشارة المتغير x ($y = + 1$ إذا كانت x موجبة و $y = 0$ إذا كانت $x = 0$ و $y = - 1$ إذا كانت x سالبة) .
SIN	10 LET Y=SIN(X)	تحسب جيب المتغير x ، $y = \sin(x)$ ، حيث x بالتقدير الدائرى
SQR	10 LET Y=SQR(X)	تحسب الجذر التربيعى للمتغير x ، $y = \sqrt{x}$ حيث $x > 0$
TAB	20 PRINT TAB(N);X	تسبب تحريك رأس الطباعة للنهاية الطرفية للآلة الكاتبة لتكون فى موضع معين . (العمود الأيسر يعتبر العمود رقم صفر) .
TAN	10 LET Y=TAN(X)	تحسب الظل للمتغير x ، $y = \tan(x)$ حيث x بالتقدير الدائرى

+ يمثل الرمز e أساس النظام الطبيعى للوغاريتمات (Naperian) وهو رقم غير منطق . وتساوى قيمته التقريبية 2.718282

استخدام كل الدوال الموجودة فى الجدول ٥ — ١ يجب أن يكون واضحاً ما عدا دالة INT ودالة TAB وسوف نناقشها فيما بعد . لاحظ أن بعض الدوال (مثل LOG ، SQR) تتطلب خلاصة موجبة . إذا كانت أى من هذه الدوال لها خلاصة سالبة سوف تتجاهل الإشارة السالبة وسوف تنفذ الحسابات على أساس القيمة المطلقة للخلاصة . ومع ذلك فدائماً تطبع رسالة خطأ عند حساب قيمة الدالة ، تشير إلى أن قيمة الخلاصة التى أعطيت للدالة كانت قيمة سالبة .

تتطلب دالة INT بعض الإيضاحات الإضافية . تسبب هذه الدالة بتر الخلاصة إذا كانت القيمة الموجبة (أى سوف نسقط من الاعتبار الجزء العشرى) . وبذلك سوف تولد دالة INT رقما صحيحاً موجباً قيمته أقل من خلاصته . ومن الناحية الأخرى إذا كانت الخلاصة لها قيمة سالبة فان دالة INT سوف تنتج رقما صحيحاً سالباً قيمته أكبر من خلاصته . وذلك موضحاً فى المثال ٥ — ٢ التالى :

مثال ٥ — ٢

فلندرس الجملة .

```
10 LET Y=INT(X)
```

إذا كانت X (الخلاصة) تمثل الرقم 12.9 ، فان القيمة التى تحدد للمتغير Y هى 12 . ومن الناحية الأخرى ، إذا كانت X تمثل 4.2 – فإن القيمة التى تحدد بها Y هى 5 – .

تمكن دالة TAB المبرمج من توصيف المكان المحدد لكل بند من قائمة بنود المخرجات فى جملة PRINT . وذلك يسمح بمرونة أكثر فى المباعدة بين بيانات المخرجات من الطرق الموصوفة فى قسم ٢ — ١٠ . كل مرة تظهر فيها دالة TAB فى قائمة مخرجات ، سوف تتحرك رأس الطباعة أو مؤشر النهاية الطرفية المركزية ناحية اليمين حتى تصل إلى العمود الموصوف (لاحظ أن العمود الموجود أقصى اليسار يعتبر العمود رقم 0) . إذا كان موضع رأس الطباعة فعلا وراء نطاق (أى إلى يمين) العمود المشار إليه ، فسوف تهمل دالة TAB .

مثال ٥ – ٣

نفرض أننا نريد طباعة القيم A و B و C على سطر واحد ، حيث تطبع القيمة الأولى ابتداء من العمود رقم 9 ، والقيمة الثانية فى العمود 29 والقيمة الثالثة فى العمود 47 . (لاحظ أن هذه الأعمدة حقيقة هى الأعمدة رقم 10th و 30th ، 48th على الترتيب). ويتم ذلك بكتابة :

```
100 PRINT TAB(9);A;TAB(29);B;TAB(47);C
```

ومن الناحية الأخرى إدرس الجملة التالية :

```
150 PRINT "NAME AND ADDRESS";TAB(12);N$
```

سوف يتم تجاهل دالة TAB فى هذه الحالة ، حيث رأس الطباعة (أو مؤشر الشاشة) تحركت إلى العمود رقم 16 بعد الانتهاء من طباعة سلسلة الحروف NAME AND ADDRESS . وعلى ذلك فإن سلسلة الحروف الممثلة بالمتغير N$ سوف تبدأ فى العمود 16 . ومع ذلك ، فإذا كانت خلاصة دالة TAB أكبر من 16 فيمكن تحريك رأس الطباعة (أو مؤشر الشاشة) إلى المكان المطلوب .

استخدام الدالة المكتبية ليس مقصورا على جملة LET أو جملة PRINT ولكن يمكن ظهور الدالة المكتبية أينما يوجد متغير عادى . وعلاوة على ذلك ، فلا يستلزم أن تكون الخلاصات متغيرات بسيطة بل يمكن استخدام الثوابت والمتغيرات ذات الأدلة والصيغ الرياضية حتى أنه يمكن الرجوع إلى دوال أخرى كخلاصات صحيحة للدالة . وسوف تؤدى عملية البتر أوتوماتيكيا عند الضرورة (للدوال التى تتطلب خلاصات ذات قيمة صحيحة (مثلا كدالة TAB)) .

مثال ٥ – ٤

كل من الجمل التالية أمثلة صحيحة لاستخدام دالة مكتبية :

```
 40 LET X1=(-B+SQR(D))/(2*A)
 60 IF ABS(X-X1)<.00001 THEN 110
100 PRINT SIN(T), COS(T), TAN(T), LOG(T), EXP(T)
200 FOR J=0 TO INT(Y)
 75 LET A=LOG(SQR(ABS(P)))
```

استخدام الدوال المكتبية مشروح فى المثال التالى . (لاحظ أن الدوال المكتبية تم استخدامها فى الأشكال ٤ – ٢ و ٤ – ١٠) .

مثال ٥ – ٥ جدول دوال A Table of Functions

نفرض أننا نريد تكوين جدول للقيم $\sin(x)$ و $\cos(x)$ و $\tan(x)$ و $\log(x)$ و e^x لعدد 21 قيمة للمتغير x التى تتراوح قيمته ما بين 0 و π . يمكن إنجاز ذلك بسهولة باستخدام حلقة FOR-TO مع البيانات المحسوبة ثم يتم عرضها فى شكل عمودى .

المخطط التمهيدى للبرنامج The Program Outline

سوف نجرى مواصلة الحسابات كما يلى :

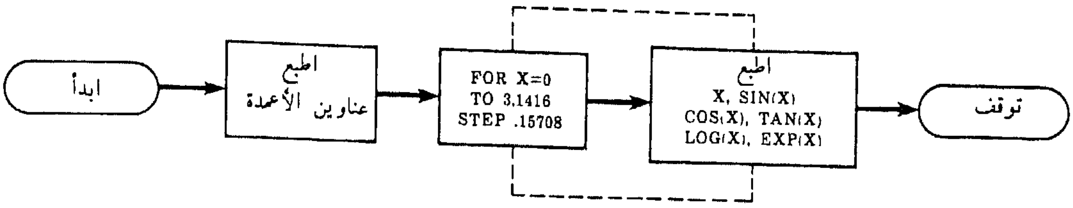
١ – اطبع عناوين الأعمدة الستة التالية :

X SIN(X) COS(X) TAN(X) LOG(X) EXP(X)

٢ – اطبع القيم x و $\sin(x)$ و $\cos(x)$ و $\tan(x)$ و $\log(x)$ و e^x لكل قيمة من قيم x تتراوح قيمها من $x = 0$ إلى $x = 3.1416$ بحجم خطوة 0.15708 (والتى تساوى $3.1416 \div 20$). وسوف تولد ذلك 21 سطراً من المخرجات .

٣ – توقف .

خريطة سير العمليات المناظرة لذلك موضحة فى شكل ٥ — ١ .



شكل ٥ — ١

برنامج البيسك **The BASIC Program**

مبين فى شكل ٥ — ٢ برنامج بيسك كامل متبوع بالبيانات المحسوبة المخرجة . لا يتطلب البرنامج أى شرح خاص ولكنه برنامج مباشر . ومع ذلك ، فيجب أن نشير إلى توخى الحرص عند تحديد خلاصات الدالة TAB حتى يتسنى طباعة أعمدة المخرجات الستة متباعدة بالتساوى على عرض الصفحة . يجب أن ينفذ البرنامج عدة مرات ، بتعديلات متتالية لخلاصات دالة TAB بعد كل مرة تنفذ فيها ، حتى يتم التوصل إلى تباعد الأعمدة بصورة صحيحة .

تبين مخرجات البيانات بوضوح القيم المحسوبة $\sin(x)$ و $\cos(x)$ و $\tan(x)$ و $\log(x)$ و e^x لكل قيمة من قيم x . لاحظ صدور رسالة خطأ عند محاولة حساب لوغاريتم للقيمة صفر (log) . ويتبع ذلك الرقم $1.70141E + 38$ — وهو أكبر رقم (فى القيمة) سالب يمكن أن يتلاءم مع سعة الحاسب . (وحقيقة فان لوغاريتم صفر يعرف بأن له قيمة سالبة لا نهائية) . كان يمكن تجنب مثل هذا الموقف بتضمين جملة IF فى البرنامج .

```
10 REM GENERATION OF A TABLE OF MATHEMATICAL FUNCTIONS
20 PRINT TAB(7);"X";TAB(15);"SIN(X)";TAB(27);"COS(X)";
30 PRINT TAB(39);"TAN(X)";TAB(51);"LOG(X)";TAB(63);"EXP(X)"
40 PRINT
50 FOR X=0 TO 3.1416 STEP .15708
60    PRINT TAB(3);X;TAB(13);SIN(X);TAB(25);COS(X);
70    PRINT TAB(37);TAN(X);TAB(49);LOG(X);TAB(61);EXP(X)
80 NEXT X
90 END

>RUN
```

```
EX5.5          09:28          15-MAR

      X        SIN(X)       COS(X)       TAN(X)       LOG(X)       EXP(X)

   0         0            1            0
% LOG OF ZERO IN LINE 70
          -1.70141E+38
                                                                   1
   0.15708   0.156435     0.987688     0.158385     -1.851         1.17009
   0.31416   0.309018     0.951056     0.324921     -1.15785       1.36911
   0.47124   0.453991     0.891006     0.509527     -0.752388      1.60198
   0.62832   0.587786     0.809016     0.726545     -0.464706      1.87446
   0.7854    0.707108     0.707105     1.           -0.241562      2.19328
   0.94248   0.809018     0.587783     1.37639      -5.92406E-2    2.56634
   1.09956   0.891008     0.453988     1.96262       9.49101E-2    3.00284
   1.25664   0.951057     0.309014     3.07771       0.228441      3.5136
   1.41372   0.987689     0.156431     6.31389       0.346225      4.11122
   1.5708    1           -3.65144E-6  -273864.       0.451585      4.81049
   1.72788   0.987688    -0.156438    -6.31359       0.546895      5.62871
   1.88496   0.951055    -0.309021    -3.07764       0.633907      6.58609
   2.04204   0.891004    -0.453995    -1.96259       0.713949      7.70631
   2.19912   0.809014    -0.587789    -1.37637       0.788057      9.01707
   2.3562    0.707103    -0.707111    -0.999989      0.85705       10.5508
   2.51328   0.587781    -0.80902     -0.726534      0.921589      12.3454
   2.67036   0.453985    -0.891009    -0.509518      0.982213      14.4452
   2.82744   0.309011    -0.951059    -0.324912      1.03937       16.9021
   2.98452   0.156428    -0.987689    -0.158377      1.09344       19.777
   3.1416   -7.25607E-6  -1            7.25607E-6    1.14473       23.1409

TIME:  0.18 SECS.
```

شكل ٥ — ٢

فى الواقع تتضمن جميع نسخ البيسك الدوال المكتبية المبينة فى الجدول ٥ — ١ وفى الملحق B . وتتضمن كثير من نسخ اللغة دوالا مكتبية إضافية ، قد يكون بعضها فريداً لهذه النسخة بالذات . ومعظمها لها طبيعة عددية لكن بعضها تقبل خلاصات حرفية أو تعطى نتائج حرفية (أو كليهما) وهذا صحيح وخصوصاً فى نسخ بيسك الحاسبات الدقيقة (انظر قسم ٩—٣) . ويجب أن يرجع القارىء إلى المرجع الخاص بجهاز معين ليحدد تماماً الدوال المكتبية المتاحة .

٥ — ٢ القوائم والجداول (المجموعات المتراصة) LISTS AND TABLES (ARRAYS)

عند كتابة برنامج كامل فغالباً من الأفضل أن نشير لمجموعة كاملة من البنود فى وقت واحد . مثل هذه المجموعة غالباً ما يشار إليها كمجموعة **متراصة** . فمثلا ، يمكننا أن نختص **بقائمة** كاملة من البنود (تعرف أيضاً **كمجموعة متراصة ذات بعد واحد**) ، أو بكل المداخل فى **جدول** (**مجموعة متراصة ذات بعدين**) . يسمح لنا البيسك بالإشارة لعناصر من القوائم والجداول كما لو كانت متغيرات طبيعية ، وبذلك يجعل تداول المجموعات المتراصة بسيطاً بقدر الإمكان .

يمكن للعناصر الموجودة فى قائمة أو جدول أن تكون إما كميات عددية أو سلاسل حرفية . (تسمح بعض نسخ بيسك بقوائم من السلاسل الحرفية ولكن لا تسمح بجداول من السلاسل الحرفية) . ومع ذلك فيجب أن تكون كل عناصر المجموعة المتراصة من نفس النوع (أى كلها عددية أو كلها سلاسل حرفية) . يجب أن تسمى المجموعة المتراصة التى تحتوى على عناصر رقية بحرف واحد ، بينما يشار إلى المجموعة المتراصة من السلاسل الحرفية بحرف تتبعه علامة $ (الدولار) . لاحظ أن أسماء المجموعات المتراصة المكونة من حرف يتبعه رقم أو حرف ثم رقم ثم علامة الدولار ، غير مسموح بها فى البيسك التقليدى .

يجب أن يكون إسم المجموعة المتراصة فريداً بداخل البرنامج ، أى لا يمكن لمجموعتين متراصتين أن تحملا نفس الإسم . ولكن يمكن لمجموعة متراصة ومتغير عادى أن يكون لهما نفس الإسم . ولكن مثل هذا التكرار فى أسماء البرنامج يمكن أن يسبب خلطاً ، وبذلك لا نوصى به .

مثال ٥ — ٦

يحتوى برنامج على قائمة من الأسماء وجدول من الأرقام . سوف يطلق على القائمة **L$** وعلى الجدول T .

يمكن أيضاً أن يتضمن البرنامج متغيراً حرفياً عادياً يسمى L$ ومتغيراً رقياً عادياً يسمى T . سوف تكون هذه المتغيرات منفصلة ومختلفة عن المجموعات المتراصة L$ و T . ولكن من المفضل أن تسمى هذه المتغيرات بأسماء مختلفة (مثال L1$ و T9) ، بذلك نتجنب أى خلط ممكن حدوثه بين المجموعات المتراصة وأى متغيرات أخرى فى البرنامج .

٥ — ٣ المتغيرات ذات الأدلة SUBSCRIPTED VARIABLES

تعرف العناصر الفردية بداخل مجموعة متراصة **كمتغيرات ذات أدلة** . ويمكن الرجوع إلى مثل هذا العنصر بذكر إسم المجموعة المتراصة تتبعه قيمة الدليل . وفى حالة الجدول فيجب توصيف دليلين منفصلين بفصلة (,) . وبذلك فان P(3) عنصر من القائمة P . و T(5, 5) عنصر من الجدول T . يجب أن تكون للأدلة قيما صحيحة موجبة ولا يمكن أن تكون قيما سالبة .

مثال ٥ — ٧

المراد وضع عشرة أسماء (سلاسل) فى قائمة فردية تسمى **L$** ، وسوف يشار للأسماء كما يلى :
L$(1) و L$ (2) .... L$ (10), . لاحظ أن الدليل يأخذ أرقاماً صحيحة تتراوح ما بين 1 إلى 10 .

وبالمثل فالمراد تنظيم أربعين رقماً فى جدول باسم T له 5 صفوف و 8 أعمدة يمثل أول دليل (رقم الصف) وسوف يأخذ قيما صحيحة تتراوح ما بين 1 إلى 5 ، يمثل الدليل الثانى (رقم العمود) وسوف يتراوح من 1 إلى 8 . وبذلك فإن الرقم الموجود فى الصف الثالث والعمود الرابع سوف يشار إليه (3, 4) T . . . الخ .

ليس من الضرورى كتابة الدليل كثابت . ولكن يمكن استخدام المتغيرات والصيغ الرياضية وأسماء الدوال أيضاً . ومع ذلك ، يجب أن تكون قيمة الدليل أما صفراً أو رقماً صحيحاً موجباً . أما إذا كان ناتج الصيغة الرياضية أو مرجع الدالة قيمة غير صحيحة للدليل ، فإن هذه القيمة سوف تبتر ، وبذلك تكون نتيجة للدليل قيمة صحيحة موجبة . أما إذا كانت النتيجة المولدة للدليل قيمة سالبة ، أو رقماً موجباً كبيراً جداً فسيتوقف التنفيذ وتطبع رسالة خطأ .

مثال ٥ – ٨

كل المتغيرات ذات الأدلة المبينة فيما بعد مكتوبة بصورة صحيحة :

P(3)	T(8,5)
P(K)	T(J1,J2)
P(C(J))	T(6,N)
P(2*A−B)	T(A1+B1, A2+B2)
P(SQR(X↑2+Y↑2))	T(ABS(X+Y), ABS(X−Y))

إدرس المتغيرات ذات الدليل (P (2*A — B نفرض أن الصيغة 2*A — B لها قيمة 2.8 ، فإن هذه القيمة سوف تبتر وسوف يفسر المتغير ذو الدليل P(2) أما إذا كانت هذه الصيغة لها القيمة 3.6 — فسوف تطبع رسالة خطأ ويتوقف تنفيذ البرنامج .

يمكن استخدام المتغيرات ذات الأدلة بداخل البرنامج بنفس الطريقة التى تستخدم بها المتغيرات العادية . والمثال التالى يوضح ذلك .

مثال ٥ – ٩ كلمة غير مرتبة Word Unscrambling

مسألة مشوقة تتضمن مداولة المتغيرات ذات الأدلة ألا وهى إعادة تنظيم مجموعة من الحروف لتكون كل الكلمات الممكنة . فلنفرض مثلا ، أنه تم إعطاؤنا أربعة حروف مثل OPST . ونود تكوين كل التوافقات الممكنة من هذه الحروف الاربعة ثم بعد ذلك الكلمات الممكنة . وبذلك يمكن إيجاد كل الكلمات ذات الحروف الأربعة التى يمكن تكوينها من الحروف الأربعة الأصلية (POST و POTS و TOPS و STOP و SPOT)

طريقة إجراء الحسابات Computational Procedure

ومن أجل عمل ذلك ، دعنا نضع الحروف الأربعة المعطاة فى قائمة ونطلق عليها L$. وسوف يمثل كل حرف بمتغير ذى دليل L$(1) و L$(2) و . . وبذا يكون الهدف هو كتابة كل التوافقات الممكنة من المتغيرات ذات الدليل .

دعنا نطبع الحروف الأربعة المعطاة بالترتيب الذى يرمز له بالمؤشرات I1 و I2 و I3 و I4 حيث I1 هو دليل أول حرف يجب طباعته ، ويشير I2 لحرف الثانى المطلوب طباعته ، وهكذا . فمثلا إذا كان I1 = 3 و I2 = 2 و I3 = 4 و I4 = 1 ، فسوف تطبع الحروف بالترتيب .

L$(3)   L$(2)   L$(4)   L$(1)

ونود أن نكتب برنامج بيسك يسمح أن تأخذ I1 و I2 و I3 و I4 كل القيم الممكنة مع التقيد بأن يأخذ كل مؤشر قيمة عددية فريدة . معنى ذلك أنه لا يسمح لأى مؤشرين أن يكون لهما نفس الرقم .

بالإضافة لجملتى INPUT و PRINT ، فإن برنامج البيسك سوف يتكون أساساً من ثلاث حلقات تكرارية متداخلة FOR–TO. أقصى حلقة خارجية تحدد قيمة I1 ، والحلقة التالية تحدد قيمة I2 ، مع عمل اختبار للتأكد أن I2 تختلف عن I1 وسوف تعطى الحلقة الداخلية قيما للمتغير I3 ، ومرة أخرى يجرى اختبار لنرى أن I3 لها قيمة مختلفة عن I1 أو I2 وأخيراً ، يمكننا الحصول على قيمة I4 بملاحظة أن مجموع الأعداد الأربعة الأولى هى 10 . وبذلك فإذا أعطيت I1 و I2 و I3 قيما منفردة ، يمكننا حساب I4 من الصيغة .

$$I4 = 10 - (I1 + I2 + I3)$$

المخطط التمهيدى للـبرنامج The Program Outline

يمكننا عمل مخطط تمهيدى للإجراء كاملا كما يلى :

١ – إقرأ L$(1) و L$(2) و L$(3) و L$(4),

٢ – نفذ التالى لقيم I1 = 1, 2, 3, 4 .

(أ) حدد قيمة للمتغير I1

(ب) نفذ التالى مع جعل I2 = 1, 2, 3, 4

(i) حدد قيمة للمتغير I2 .

(ii) أجر اختباراً لترى أن I2 تختلف عن I1 ، إذا لم يتحقق ذلك فأضف مقدار الوحدة إلى I2 وأعد الاختبار مرة أخرى.

(iii) لكل قيمة من I1 و I2 اعمل الآتى مع جعل I3 = 1, 2, 3, 4

(١) حدد قيمة I3

(٢) أجر اختباراً لترى أن I3 تختلف عن كل من I1 و I2 . إذا لم يتحقق ذلك أضف مقدار الوحدة إلى I3 . وأعد الاختبار مرة أخرى .

(٣) احسب قيمة I4 من الصيغة $I4 = 10 - (I1 + I2 + I3)$

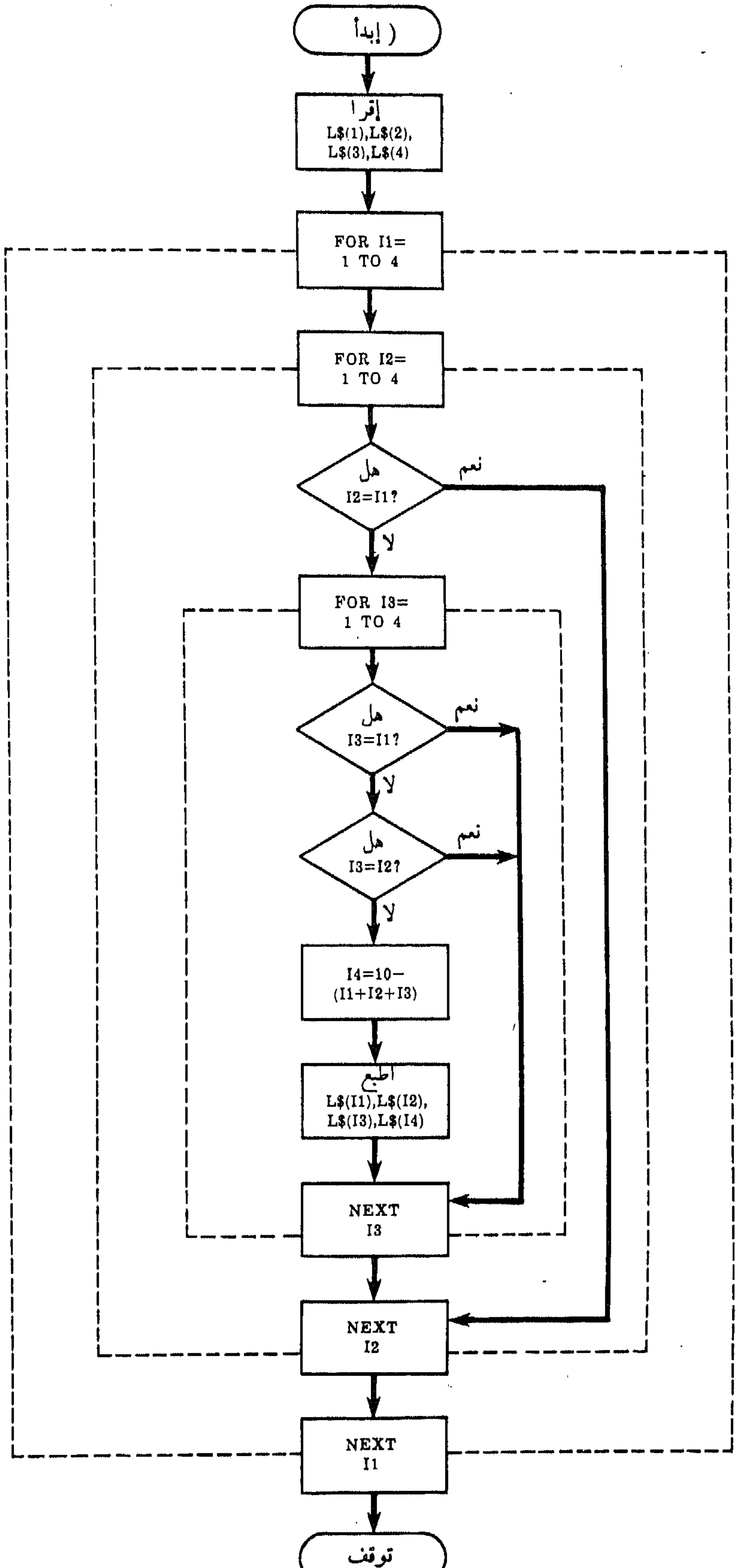
(٤) اطبع L$(I1) و L$(I1) و L$(I3) و L$(I4) ،

٣ – توقف .

خريطة سير العمليات لهذا الإجراء مبينة فى شكل ٥ – ٣ .

برنـامج البيسك The BASIC Program

يظهر برنامج البيسك الحقيقى فى شكل ٥ – ٤ . ومبين تحت البرنامج المخرجات المولدة لحروف الأربعة OPST . نرى أن هناك 24 طريقة مختلفة يمكن بها توافق الحروف الأربعة . ويمكن برهنة ذلك رياضياً بأن ذلك هو العدد الصحيح للتوافقيات . وبفحص مرئى يمكن إيجاد الكلمات الخمس التى يمكن التعرف عليها وهى : POST و POTS و SPOT و STOP و TOPS . وقد تم إحاطة هذه الكلمات بدوائر فى شكل ٥ – ٤ .


إبدأ
إقرأ
L$(1),L$(2),
L$(3),L$(4)
FOR I1=
1 TO 4
FOR I2=
1 TO 4
هل
I2=I1?
نعم
لا
FOR I3=
1 TO 4
هل
I3=I1?
نعم
لا
هل
I3=I2?
نعم
لا
I4=10−
(I1+I2+I3)
اطبع
L$(I1),L$(I2),
L$(I3),L$(I4)
NEXT
I3
NEXT
I2
NEXT
I1
توقف

شكل ٥ – ٣

```
10   REM FØUR-LETTER WØRD UNSCRAMBLER
20   PRINT "TYPE ANY FØUR LETTERS:"
30   PRINT
40   INPUT L$(1),L$(2),L$(3),L$(4)
50   PRINT
60   FØR I1=1 TØ 4
70      FØR I2=1 TØ 4
80         IF I2=I1 THEN 150
90         FØR I3=1 TØ 4
100           IF I3=I1 THEN 140
110           IF I3=I2 THEN 140
120           LET I4=10-(I1+I2+I3)
130           PRINT L$(I1);L$(I2);L$(I3);L$(I4)
140        NEXT I3
150     NEXT I2
160 NEXT I1
170 END

>RUN

TYPE ANY FØUR LETTERS:

 ?Ø,P,S,T

ØPST
ØPTS
ØSPT
ØSTP
ØTPS
ØTSP
PØST
PØTS
PSØT
PSTØ
PTØS
PTSØ
SØPT
SØTP
SPØT
SPTØ
STØP
STPØ
TØPS
TØSP
TPØS
TPSØ
TSØP
TSPØ
```

شكل ٥ — ٤

٥ — ٤ تعريف المجموعات المتراصة ــ جملة DIM :

DEFINING ARRAYS – THE DIM STATEMENT

يحدد البيسك 11 عنصراً لكل قائمة و 121 عنصراً (11 صفاً و 11 عموداً) لكل جدول يظهر فى البرنامج أوتوماتيكياً . وبذلك يسمح لكل دليل أن يتراوح ما بين صفر إلى 10 . وبالطبع قد لا يتطلب الأمر الاستفادة من كل العناصر الموجودة فى المجموعة المتراصة — فيمكن بدء الدليل بقيمة صحيحة أكبر من الصفر أنهاءه بقيمة صحيحة أقل من 10. (فى مثال ٥—٩ ، مثلا ، يتراوح الدليل من 1 إلى 4 فقط) .

ويمكننا أيضاً الاستفادة من مجموعات متراصة أكبر من ذلك . ولعمل ذلك يجب تعريف حجم كل مجموعة متراصة ، أى ، يجب توصيف أقصى عدد من العناصر فى كل مجموعة متراصة . ويمكن إنجاز ذلك بواسطة جملة *DIM* (أبعاد *DIMENSION*) .

تتكون جملة DIM من رقم الجملة ، تتبعها الكلمة الدالة DIM يتبعها اسم مجموعة مترّاصة أو أكثر يفصل بينها فصلة (,) . يجب أن يتبع كل اسم لمجموعة مترّاصة رقم ثابت صحيح أو رقمان (رقم واحد للقائمة ورقمان للجدول) محصورة بين قوسين ، وعند استخدام رقمين صحيحين يجب أن نفصل بينهما بفصلة (,) . وتشير هذه الأرقام لأقصى قيمة مسموح بها للدليل فى المجموعة المترّاصة

مثال ٥ – ١٠

يحتوى برنامج على جدول يسمى A وقائمتين رقيتين B و C وقائمة سلسلة حرفية تسمى F$. مطلوب أن يكون الجدول 50 صفا و 100 عموداً وسوف تحتوى القائمتان B و C على 100 عنصر و 50 عنصراً على الترتيب وسوف تحتوى F$ على 65 عنصراً فيمكن للبرنامج فى هذه الحالة أن يحتوى على جملة DIM التالية :

```
20  DIM  A(50,100), B(100), C(50), F$(65)
```

تحجز هذه الجملة حقيقة 51 صفا و 101 عموداً للجدول A و 101 عنصراً للقائمة B و 51 عنصراً للقائمة C و 66 عنصراً للقائمة F $. وبذلك فإن المجموعات المتراصة تكون اكبر قليلا من المطلوب . ولكن كثيرا من المبرمجين يفضلون أن يبدأ الدليل بالقيمة 1 وليس بالقيمة صفر ، كما لو كانوا يتعاملون مع متغيرات جبرية لها أدلة . وتحت هذه الظروف يكون من الأسهل كتابة جملة DIM بالصورة المبينة عالياً .

يمكن أن تظهر جملة DIM فى أى مكان فى برنامج البيسك ولكن من الأفضل عملياً أن تضع جملة DIM عند بداية البرنامج حتى يكون وجودها واضحاً . ويسمح ذلك للمبرمج أو المستفيد أن يحدد أقصى أحجام للمجموعات المتراصة بسهولة وبسرعة .

يمكن أن تتضمن جملة DIM القوائم التى تحتوى على أقل من 11 عنصراً أو الجداول التى تحتوى على أقل من 121 عنصراً ، بالرغم من عدم أهمية عمل ذلك . ويترتب على ذلك حجز كلمات أقل فى ذاكرة الحاسب إلا أنه يتطلب بعض الحرص عند التعامل مع الجداول ، لأن أحد الأدلة يمكن أن يتجاوز القيمة 10 حتى لو كان العدد الإجمالى للعناصر أقل من 121 وعند حدوث ذلك يجب استخدام جملة DIM وسوف نرى مثل هذا الموقف فى المثال التالى .

مثال ٥ – ١١

برنامج يحتوى على جملة DIM التالية :

```
30  DIM  P(6), Q(10), R(5,15)
```

ويترتب على ذلك حجز 7 كلمات من المخزن للعناصر الستة للقائمة P و 11 كلمة للقائمة Q و 96 كلمة (6 صفوف 16 عموداً)للجدول R .

إقحام P و Q فى جملة DIM كان غير مهم حقيقة ، حيث يتم تحديد عدد كاف من المخازن أوتوماتيكياً . ومن الناحية الأخرى ، كان يجب أن نضمن الجدول R فى جملة DIM حيث أن الدليل الثانى يتعدى 10 وهذا صحيح بالرغم من أن العدد الإجمالى للكلمات المطلوبة بواسطة R أقل من 121 عنصراً .

وسوف نرى مثالا كاملا لبرنامج يتطلب وجود جملة DIM فى جزء لاحق من هذا الفصل .

٥ – ٥ ادخال بيانات الادخال – جملتى READ و DATA

ENTERING INPUT DATA—THE READ AND DATA STATEMENTS

تتطلب عدة برامج بيسك أن تدخل للحاسب كمية كبيرة من عناصر البيانات ، ويمكن إنجاز ذلك بجملة INPUT ، على الرغم من أن تنفيذ ذلك أمر مرهق . وعادة يكون أكثر ملاءمة إدخال مثل هذه البيانات بواسطة جملتى *READ* و *DATA* . تستخدم هاتان الجملتان أيضاً لإدخال البيانات للبرنامج الذى يستخدم أسلوب التشغيل بالدفعات وليس بأسلوب التشغيل بالمشاركة الزمنية .

توصف جملة READ المتغيرات التى يجب أن تدخل قيمها للحاسب . وتتكون هذه الجملة من رقم جملة ، تتبعها الكلمة الدالة READ ، تتبعها قائمة بمتغيرات الإدخال . يمكن أن تحتوى الجملة على كل من المتغيرات العادية والمتغيرات ذات الأدلة ممثلة قيما عددية (أو / و) قيما حرفية . إذا احتوت الجملة على متغيرين أو أكثر فيجب أن تفصل بينهما الفصلة (,) .

والغرض من جملة DATA هو تحديد قيم مناسبة للمتغيرات التى سبق وأن ظهرت فى جملة READ . وتتكون جملة DATA من رقم جملة تتبعها الكلمة الدالة DATA ، تتبعها مجموعة من الأرقام (أو / و) السلاسل الحرفية ، تفصل بينها الفصلة (,) . كل رقم (أو / و) سلسلة حرفية فى جملة DATA يجب أن يناظر متغيراً من نفس النوع فى جملة READ .

مثال ٥ – ١٢

يحتوى برنامج بيسك على الجمل التالية :

```
30 READ K,N$,Z(1)
   ...
120 DATA 12,SEVENTEEN,-5
```

يترتب على هذه الجملة أن نعطى المتغير K القيمة 12 والمتغير N$ السلسلة SEVENTEEN والمتغير Z(1) بالقيمة 5 – .

لا يستلزم أن تناظر جملة DATA لكل جملة READ معينة ، برغم أن مثل هذا التناظر جملة تقابل جملة مسموحاً به فى حالة الرغبة فى ذلك (كما رأينا فى المثال السابق) . ولكن النقطة الهامة هى أن كل جمل DATA الموجودة فى البرنامج تتجمع وتكون كتلة من قيم البيانات . ويجب أن يناظر كل عنصر فى كتلة البيانات متغيراً فى جملة READ . يجب أن يكون هذا التناظر بالنسبة لكل من الترتيب والنوع .

مثال ٥ – ١٣

يحتوى برنامج بيسك على الجمل التالية :

```
 40 READ A,B,C
 50 READ P$,Q$
 60 READ F(1),F(2),F(3)
    ...
210 DATA 3,-2,11,AB
220 DATA CD,-8,0,10
```

يترتب على هذه الجمل أن تحدد قيم A و B و C بالأرقام 3 و 2 – و 11 . وتحدد قيم P$ و Q$ بالحروف AB و CD وتحدد قيم المتغيرات ذات الأدلة F (1) و F(2) و F (3) بالأرقام 10 و 0 و 8 – . لاحظ أن جمل READ لاتناظر جمل DATA على أساس جملة ، لكن تناظر المتغيرات فى جمل READ العناصر فى كتلة البيانات من حيث الترتيب والنوع .

وكان يمكن أيضاً جمع جملتى DATA لتكون جملة واحدة .

```
210 DATA 3,-2,11,AB,CD,-8,0,10
```

وسوف تكون النتائج هى نفسها التى عرضت عالية .

يجب أن نشير إلى أن البيانات التى تم توصيفها خلال جملتى READ و DATA هى جزء من البرنامج ، بعكس البيانات التى تم إدخالها خلال جمل INPUT ، ولذلك فإن البيانات التى تحتويها كتلة البيانات تخزن حينما يخزن البرنامج ، ويتم تحديد قيمها لمجموعة مناسبة من المتغيرات حين يتم تنفيذ البرنامج . وعلى ذلك تكون هذه البيانات دائمة نسبياً ، ويمكن تغييرها فقط بواسطة تعديل جملة DATA أو أكثر بداخل البرنامج .

يجب ملاحظة القواعد التالية عند وضع بنود البيانات فى كتلة البيانات :

١ – يجب أن تناظر بنود البيانات قائمة المتغيرات فى جمل READ من حيث الترتيب والنوع . ويجب أن يكون عدد العناصر الموجودة فى كتلة البيانات مساوياً لعدد المتغيرات فى جمل READ وعناصر البيانات الزائدة سوف يتم تجاهلها .

٢ — يجب أن تفصل عناصر البيانات بداخل جملة DATA بواسطة فصلة (,) ولكن يجب ألا يتبع آخر عنصر من البيانات فى جملة DATA فصلة (,) .

٣ — يجب أن تتكون عناصر البيانات من أرقام أو سلاسل حرفية . وغير مسموح بالمتغيرات أو الصيغ الرياضية .

٤ — يجب أن تنحصر السلاسل الحرفية التى تحتوى فصلات (,) أو تبدأ بفراغ بين علامتى اقتباس . ويمكن أن تنحصر السلاسل الحرفية الأخرى بين علامتى الاقتباس إذا أردت ذلك .

يمكن لجمل DATA أن تظهر فى أى مكان من برنامج البيسك ولكنه ، تقليد جيد أن توضع كل جمل DATA متتالية بقرب نهاية البرنامج . وبذلك يكون مكان وتكوين كتلة البيانات واضحاً للغاية .

يوضح المثال التالى تحديد مجموعة من المتغيرات ذات الأدلة بقيم عددية وكيفية هذه المتغيرات ذات الأدلة . وقد تم أيضاً شرح استخدام جملة DIM .

مثال ٥ — ١٤ إعادة ترتيب قائمة من الارقام Reordering a List of Numbers

إدرس المسألة الشهيرة وهى إعادة تنظيم قائمة طولها N من الأرقام إلى توال من الأرقام المتزايدة القيمة . يجب كتابة البرنامج بطريقة تسمح بعدم استخدام مخازن أزيد من اللازم . ولذلك فسوف يحتوى البرنامج على مجموعة متراصة واحدة فقط ، حيث يعاد تنظيم عنصر واحد فى المرة الواحدة .

طريقة إجراء الحسابات Computational Procedure

سوف يفحص الإجراء القائمة بأكلها بحثاً عن أصغر رقم ويبدل هذا الرقم بأول رقم فى القائمة . بعد ذلك تفحص الأرقام N — 1 المتبقية بحثاً عن أصغر هذه الأرقام ويبدل هذا الرقم بالرقم الثانى فى القائمة.ثم بعد ذلك تفحص الأرقام N — 2 بحثاً عن أصغر رقم ، ويبدل هذا الرقم بالرقم الثالث فى القائمة ، . . وهكذا ، حتى يتم إعادة تنظيم القائمة بأكلها . يتطلب ذلك عدداً من اللفات خلال القائمة مقداره N — 1 بالرغم من أن القائمة تتناقص رقماً فى كل بحث متعاقب .

من أجل الحصول على أصغر رقم فى كل لفة ، فإننا نقارن كل رقم فى القائمة بالرقم الذى بدأنا به (ذى الترتيب I) على التوالى . إذا كان الرقم ذو الترتيب *i* أكبر من الرقم ذى الترتيب *j* مثلا ، فإننا نبدل الرقمين ، وإلا نترك الرقمين فى أماكنهما الأصلية . فور إتمام تطبيق هذا الإجراء على القائمة بأكلها ، فإن أول رقم فى القائمة سوف يكون أصغر رقم . بعد ذلك نعيد الإجراء بأكمله عدد (N — 2) من المرات ، بعدد إجمالى N — 1 من المرات (($i = 1, 2 \ldots\ldots N - 1$) .

السؤال المتبقى الوحيد هو كيف يتم تبديل الرقم ذى الترتيب *i* مع الرقم ذى الترتيب *j* . أولا « نجنب » الرقم ذا الترتيب *i* للرجوع إليه مستقبلا ، وذلك يعنى ، أننا نحتفظ بالقيمة الأصلية للمتغير ذى الدليل *i* . ثم نحدد قيمة المتغير ذى الدليل *i* بالمتغير ذى الدليل *j* . وأخيراً ، فإننا نحدد قيمة المتغير ذى الدليل *j* بقيمة المتغير ذى الدليل *i* الأصلية والتى سبق تجنيبها سابقاً . وبذا يكون قد تم التبديل المطلوب .

المخطط التمهيدى للبرنـــامج The Program Outline

يمكن القيام بتنفيذ الحسابات بالكامل بواسطة حلقتين FOR-TO ولنر كيف يمكن إنجاز ذلك ، عليك بدراسة التخطيط التمهيدى التالى :

١ — إقرأ حجم القائمة (N) .

٢ — إقرأ ثم اطبع قائمة بها عدد N من الأرقام الصحيحة الثابتة

٣ – نفذ الحسابات التالية عدد (N — 1) من المرات ، مع جعل ($i = 1, 2, \ldots\ldots, N - 1$)
قارن الرقم ذا الترتيب i وسوف يكون أول رقم فى القائمة ، مع كل رقم لاحق [أى كل رقم ذى ترتيب j ،
حيث ($j = i + 1, i + 2, \ldots N$) وعندما يكون ذو الترتيب i أكبر من الرقم ذى الترتيب j ، نستبدل الرقين .

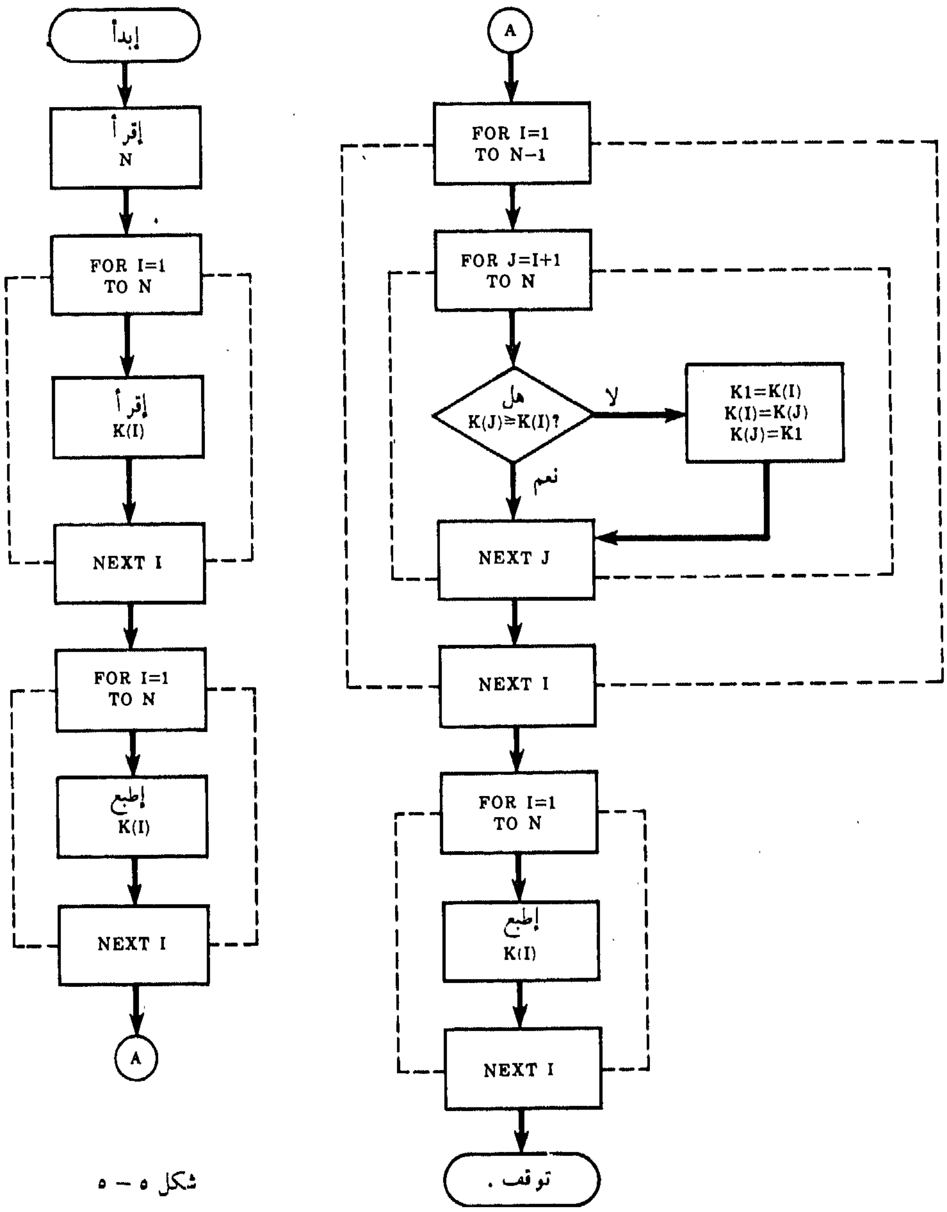
٤ – إطبع قائمة الأرقام بعد إعادة تنظيمها .

٥ – توقف .

يوضح الشكل ٥ – ٥ خريطة لسير العمليات لهذا الإجراء .

برنامج البيسك The BASIC Program

عند كتابة برنامج البيسك الفعلى دعنا نطلق اسم K على الأرقام الصحيحة وسوف نفرض أن K لن تتكون من أكثر من 101 عنصراً وبذلك فسوف نجعل أقصى قيمة مسموح بها للأدلة هى 100 .



شكل ٥ – ٥

برنامج البيسك موضح فى شكل ٥ — ٦ . لاحظ جملة DIM بقرب بداية البرنامج (سطر 20) ، وهى تحدد إن K يمكن أن تحتوى عدداً من العناصر يصل إلى 101 عنصراً . ولاحظ أيضاً استخدام حلقات FOR-TO من أجل قراءة وطباعة عناصر K (مثال ، أنظر السطور 40 إلى 60 و 90 إلى 110 و 330 إلى 350) .

يمكن القيام بإعادة التنظيم بواسطة حلقتين من حلقات FOR-TO كما سبق وصفها (السطور 140 إلى 290) . تضمن البرنامج ثلاث جمل REM . وتساعد هذه الجمل فى توضيح منطق الحلقتين FOR-TO .

وأخيراً ، لاحظ وجود جملتى DATA قرب نهاية البرنامج (السطران 360 و 370) محتوية على 21 قيمة . أولى هذه القيم (الرقم 20) وهى قيمة N ، تشير إلى أن K سيكون بها 20 عنصراً فى هذا المثال . وتتبع هذا الرقم القيم العشرون الموجودة فى K ، بترتيبها الطبيعى (أى 595 = K(1) و 78 = K(2) و 1505 = K(3) و ...... ;K (20) = 710).

تظهر المخرجات التى يولدها هذا المثال عند نهاية شكل ٥ — ٦ . نرى القائمة الأصلية للأرقام فى أول كشف المخرجات ، وتتبعها فوراً قائمة بالأرقام التى تم إعادة تنظيمها بعد ذلك .

```
10 REM PROGRAM TO REARRANGE A LIST OF NUMBERS INTO ASCENDING ORDER
20 DIM K(100)
30 READ N
40 FOR I=1 TO N
50    READ K(I)
60 NEXT I
70 PRINT "ORIGINAL LIST OF NUMBERS:"
80 PRINT
90 FOR I=1 TO N
100   PRINT K(I);
110 NEXT I
120 PRINT
130
140 REM          REPEAT INTERCHANGE N-1 TIMES
150
160 FOR I=1 TO N-1
170
180    REM       FIND SMALLEST NUMBER IN LIST AND INTERCHANGE
190
200    FOR J=I+1 TO N
210       IF K(J)>=K(I) THEN 280
220
230       REM   INTERCHANGE K(J) AND K(I)
240
250       LET K1=K(I)
260       LET K(I)=K(J)
270       LET K(J)=K1
280    NEXT J
290 NEXT I
300 PRINT
310 PRINT "REORDERED LIST OF NUMBERS:"
320 PRINT
330 FOR I=1 TO N
340    PRINT K(I);
350 NEXT I
360 DATA 20,595,78,1505,891,29,7,18,191,36,68,7051,509,212,46,726,1806
370 DATA 289,401,1488,710
380 END
>RUN

EXS.14          09:32          15-MAR

ORIGINAL LIST OF NUMBERS:

 595  78  1505  891  29  7  18  191  36  68  7051  509  212  46  726
 1806  289  401  1488  710

REORDERED LIST OF NUMBERS:

 7  18  29  36  46  68  78  191  212  289  401  509  595  710  726
 891  1488  1505  1806  7051

TIME:  0.09 SECS.
```

شكل ٥ — ٦

تتم معالجة عناصر الجدول بطريقة تشبه كثيراً نفس الطريقة التى تتم بها معالجة عناصر القائمة . يتطلب دائماً حلقتان من حلقات FOR-TO لعمليات الإدخال / الإخراج التى تجرى عند معالجة عناصر الجدول . وسترى توضيحاً لذلك فى المثال التالى .

مثال ٥ – ١٥ معالجة عناصر الجدول Table Manipulation

إدرس جدول الأرقام المبين فى الجدول ٥ – ٢ . ولنفرض أننا نرغب فى تجميع كل العناصر الموجودة فى كل صف من الجدول ، كذا كل العناصر الموجودة فى كل عمود ، دعنا نكتب برنامج بيسك يسمح لنا بالقيام بهذه الحسابات الأولية .

جدول ٥ – ٢

6	0	−12	4	17	21
−8	15	5	5	−18	0
11	3	1	−17	12	7
13	2	13	−9	24	4
−27	−3	0	14	8	−10

المخطط التمهيدى للبرنامج The Program Outline

سوف نواصل عمل الحسابات طبقاً للتخطيط التالى :

١ – إقرأ القيم التى تشير إلى عدد الصفوف M وعدد الأعمدة N .

٢ – إقرأ عناصر الجدول (T) على أساس صف بصف بعمل التالى ، لكل قيمة من قيم عداد الصفوف I الذى تتراوح قيمة مابين 1 إلى M .

لكل قيمة I ، اجعل J (عداد الأعمدة) يتراوح مابين 1 إلى N . إقرأ قيمة للعنصر T (I, J) لكل قيمة من قيم I و J .

٣ – إطبع عناصر T على أساس صف بصف . إذن فلكل قيمة I تتراوح مابين 1 إلى M نتبع مايلى :

لكل قيمة من I إجعل J تتراوح مابين 1 إلى N إطبع قيمة للعنصر T(I, J) لكل قيمة من قيم I و J .

٤ – احسب ثم اطبع مجموع عناصر كل صف ، ثم المجموع لمجاميع الصفوف (أى المجموع التراكى لكل الصفوف (كما يلى :

(أ) اجعل القيمة المبدئية للمجموع التراكى (S1) صفرا .

(ب) نفذ مايلى بالنسبة لكل قيمة من قيم I التى تتراوح ما بين 1 إلى M :

(i) اجعل القيمة المبدئية لمجموع الصف S(I) صفراً .

(ii) لكل قيمة من قيم J التى تتراوح مابين 1 إلى N أضف قيمة T(I, J) إلى قيمة S(I) ، أى ، اجعل :

S(J)=S(J)+T(I,J)

(iii) اطبع القيمة الحالية I والقيمة المناظرة S(I) .

(iv) أضف القيمة S(I) للقيمة S1 ، أى ، اجعل :

S1=S1+S(J)

(ح) اطبع القيمة النهائية S1 (أى المجموع التراكى لكل المجاميع التى سبق حسابها للصفوف) .

٥ - احسب ثم اطبع مجموع العناصر فى كل عمود و مجموع مجاميع كل الأعمدة المفردة (أى المجموع التراكى لكل الأعمدة) كمايلى :

(أ) اجعل القيمة الأولية للمجموع التراكى S1 مساوياً لصفر .

(ب) اعمل الآتى لكل قيمة من قيم J التى تتراوح من 1 إلى N .

(i) اجعل القيمة المبدئية لمجموع العمود ، (S(J ، مساوياً لصفر .

(ii) لكل قيمة من I التى تتراوح مابين 1 إلى M ، أضف قيمة (T(I, J إلى قيمة (S(J أى ، اجعل :

$$S(J) = S(J) + T(I, J)$$

(iii) اطبع القيمة الحالية J والقيمة المناظرة (S(J .

(iv) أضف قيمة (S(J إلى S1 ، أى اجعل :

$$S1 = S1 + S(J)$$

(حـ) اطبع القيمة النهائية S1 (أى ، المجموع التراكى لمجموع كل الحسابات السابقة لمجاميع الأعمدة) . لاحظ أن المجموع التراكى للاعمدة يجب أن يساوى المجموع التراكى للصفوف ، إذن يجب أن تكون قيمة S1 مساوية للقيمة المطبوعة فى الخطوة رقم 4 . ويستخدم ذلك كاختبار .

٦ - توقف .

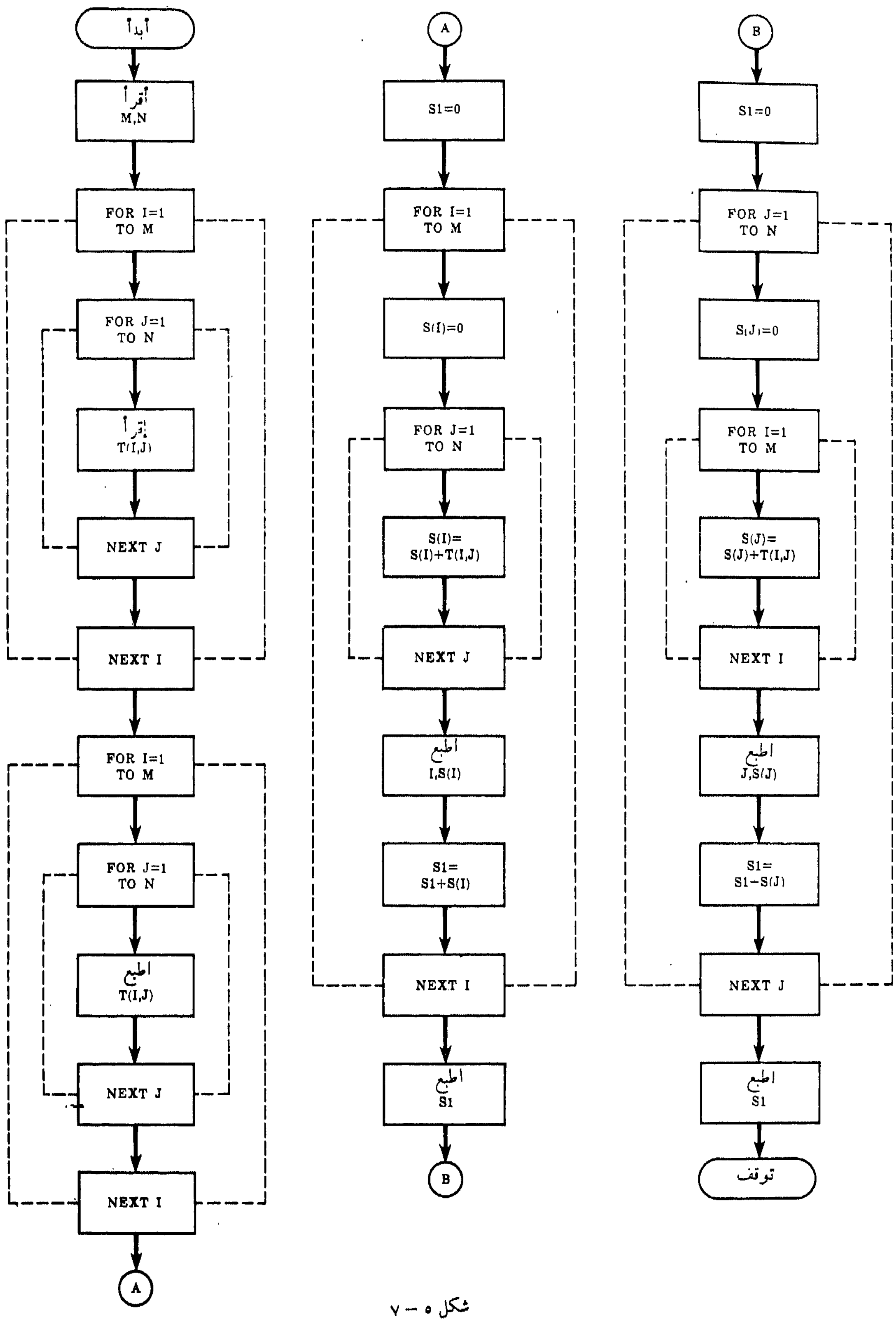
يحتوى شكل ٥ - ٧ على خريطة سير العمليات التى تناظر التخطيط السابق .

برنـــامج البيسك The BASIC Program

يقدم شكل ٥ - ٨ برنامج بيسك كاملا لهذه المسألة . لاحظ الحلقات FOR-TO المتداخلة التى تستخدم لقراءة وطباعة عناصر الجدول (أنظر السطور 70 إلى 110 و 170 إلى 220)) . استخدمت أيضاً حلقتان FOR-TO لتكوين مجاميع الصفوف (السطور 290 إلى 330) ومجاميع الأعمدة (السطور 460 إلى 540) .

تكون جملتا DATA قرب نهاية البرنامج (السطور 580 و 590) كتلة بيانات بها 32 عنصراً . الرقمان الأولان هما قيمتا M و N ، حيث تشير فى هذا المثال إلى أن الجدول يحتوى على 5 صفوف و 6 أعمدة . والأرقام المتبقية وعددها 30 هى قيم عناصر الجدول T ، على أساس صف بصف (وذلك يتطلب زيادة الدليل الثانى J بسرعة عند قراءة البيانات) .

نرى أن البرنامج لايتضمن DIM حيث لاتتعدى قيم الأدلة 10. إذا تطلب الأمر زيادة حجم المجموعة المتراصة ، تكون جملة DIM ، بالطبع ، مهمة . وأخيراً ، لاحظ أن البرنامج يحتوى على عدد من جمل REM وعدد من الأسطر الخالية ، التى تجعل قراءتها أسهل وتعطى بعض التوضيحات لمنطق البرنامج .


ابدأ
اقرأ M,N
FOR I=1 TO M
FOR J=1 TO N
اقرأ T(I,J)
NEXT J
NEXT I
FOR I=1 TO M
FOR J=1 TO N
اطبع T(I,J)
NEXT J
NEXT I
A
A
S1=0
FOR I=1 TO M
S(I)=0
FOR J=1 TO N
S(I)= S(I)+T(I,J)
NEXT J
اطبع I,S(I)
S1= S1+S(I)
NEXT I
اطبع S1
B
B
S1=0
FOR J=1 TO N
S(J)=0
FOR I=1 TO M
S(J)= S(J)+T(I,J)
NEXT I
اطبع J,S(J)
S1= S1-S(J)
NEXT J
اطبع S1
توقف

شكل ٥ — ٧

```
10 REM PROGRAM TO SUM ROWS AND COLUMNS OF A TABLE
20
30 READ M,N
40
50 REM READ ELEMENTS OF TABLE
60
70 FOR I=1 TO M
80    FOR J=1 TO N
90       READ T(I,J)
100   NEXT J
110 NEXT I
120
130 REM PRINT TABLE
140
150 PRINT "GIVEN TABLE:"
160 PRINT
170 FOR I=1 TO M
180    FOR J=1 TO N
190      PRINT T(I,J);
200    NEXT J
210    PRINT
220 NEXT I
230 PRINT
240
250 REM SUM ACROSS EACH ROW
260
270 PRINT "SUM OF COLUMNS IN EACH ROW:"
280 PRINT
290 LET S1=0
300 FOR I=1 TO M
310    LET S(I)=0
320    FOR J=1 TO N
330       LET S(I)=S(I)+T(I,J)
340    NEXT J
350    PRINT "ROW ";I,"SUM=";S(I)
360    LET S1=S1+S(I)
370 NEXT I
380 PRINT
390 PRINT "SUM OF ROW SUMS=";S1
400 PRINT
410
420 REM SUM DOWN EACH COLUMN
430
440 PRINT "SUM OF ROWS IN EACH COLUMN:"
450 PRINT
460 LET S1=0
470 FOR J=1 TO N
480    LET S(J)=0
490    FOR I=1 TO M
500       LET S(J)=S(J)+T(I,J)
510    NEXT I
520    PRINT "COLUMN ";J,"SUM=";S(J)
530    LET S1=S1+S(J)
540 NEXT J
550 PRINT
560 PRINT "SUM OF COLUMN SUMS=";S1
570
580 DATA 5,6,6,0,-12,4,17,21,-8,15,5,5,-18,0,11,3,1,-17,12,7
590 DATA 13,2,13,-9,24,4,-27,-3,0,14,8,-10
600 END
```

شكل ٥ — ٨

نرى فى شكل ٥ — ٩ المخرجات التى تم توليدها بواسطة هذا المثال . يطبع الجدول المعطى أولا ، يتبعه مجموع الصفوف (عناصر) لكل صف والمجموع التراكمى لمجاميع الصفوف . ويتبع هذه القيم مجاميع الأعمدة المختلفة والمجموع التراكمى للأعمدة . لاحظ أن المجموع التراكمى للصفوف مساو للمجموع التراكمى للأعمدة (81) ، كما هو متوقع .

```
GIVEN TABLE:

  6   0 -12   4  17  21
 -8  15   5   5 -18   0
 11   3   1 -17  12   7
 13   2  13  -9  24   4
-27  -3   0  14   8 -10

SUM OF COLUMNS IN EACH ROW:

ROW  1         SUM= 36
ROW  2         SUM=-1
ROW  3         SUM= 17
ROW  4         SUM= 47
ROW  5         SUM=-18

SUM OF ROW SUMS= 81

SUM OF ROWS IN EACH COLUMN:

COLUMN  1      SUM=-5
COLUMN  2      SUM= 17
COLUMN  3      SUM= 7
COLUMN  4      SUM=-3
COLUMN  5      SUM= 43
COLUMN  6      SUM= 22

SUM OF COLUMN SUMS= 81
```

شكل ٥ — ٩

٥ — ٦ اعادة قراءة البيانات — جملة RESTORE

REREADING DATA—THE RESTORE STATEMENT

رأينا فى القسم السابق أنه يجب الاحتفاظ دائماً بالتناظر بين المتغيرات التى تقرأ قيمها (أى قائمة المتغيرات فى جمل READ) وعناصر البيانات الفردية فى كتلة البيانات (الأرقام والحروف فى جمل DATA) . ويتم إنجاز ذلك بواسطة « مؤشرات » داخلية حيث تشير إلى العنصر التالى المطلوب قراءته من البيانات . وفى الحقيقة ، يحتفظ بمؤشرين إذا كانت كتلة البيانات تحتوى على بيانات عددية وحرفية — أحد هذين المؤشرين خاص بالأرقام (الثوابت العددية) والآخر للحروف . فى كل مرة تم قراءة عنصر من البيانات ، يتقدم المؤشر أوتوماتيكياً لعنصر البيانات التالى الذى له نفس النوع .

توجد أنواع معينة من المسائل تتطلب قراءة بعض وربما كل العناصر أكثر من مرة . ولعمل ذلك يجب إعادة أحد المؤشرين أو كليهما إلى بداية كتلة البيانات . وتستخدم **جملة *RESTORE*** لهذا الغرض .

وتتكون جملة RESTORE من رقم جملة ، تتبعها الكلمة الدالة RESTORE . ويسبب ظهور هذه الجملة إعادة كل من المؤشرين إلى أول عنصر من البيانات من النوع المناظر له بداخل كتلة البيانات .

مثال ٥ — ١٦

يحتوى برنامج بيسك على الجمل التالية :

```
30 READ A,B,C
   . . .
60 RESTORE
70 READ W,X,Y,Z
   . . .
200 DATA 1,3,5,7,9,11,13
```

تسبب الجملة 30 تحديد المتغير A بالقيمة 1 والمتغير B بالقيمة 3 والمتغير C بالقيمة 5 . وعند مصادفة الجملة رقم 60 فسوف يعود المؤشر إلى أول رقم فى كتلة البيانات . إذن فالجملة رقم 70 تسبب تحديد المتغير W بالقيمة 1 والمتغير X بالقيمة 3 والمتغير Y بالقيمة 5 والمتغير Z بالقيمة 7 . إذا كانت جملة RESTORE غير موجودة فإن W سوف تحدد بالقيمة 7 و X بالقيمة 9 ، ... وهكذا .

لاحظ أننا اهتممنا بمؤشر واحد فقط فى هذا المثال حيث أن كتلة البيانات تحتوى على ثوابت رقية فقط .

يُسمح للكلمة الدالة RESTORE أن تتبعها علامة (*) أو علامة ( ) ، وبعض نسخ البيسك عند وجود (*) فإن مؤشر الأرقام فقط هو الذى يعاد مكانه ، بينما يعاد مؤشر سلاسل الحروف إلى مكانه فى حالة وجود ($) فقط . بينما لا يسمح بوجود كل من (*) ، ($) معاً فى نفس الجملة .

مثال ٥ - ١٧

يحتوى برنامج بيسك على الجمل التالية :

```
 50 READ A,B,M$,N$
     ...
150 RESTORE*
160 READ C1,C2,F1$,F2$
     ...
200 DATA 2,4,RED,GREEN,6,8,BLUE,WHITE
```

تسبب الجملة 50 تحديد المتغيرين A و B بالقيم 2 و 4 والمتغيرين M$ و N$ بالحروف RED و GEERN . أعيد المؤشر الرقى بواسطة الجملة 150 ومن ثم تسبب الجملة 160 تحديد المتغيرين C1 و C2 بالقيم 2 و 4 ، بينما سيتم تمثيل المتغيرات F1$ و F2$ بالحروف BLUE و WHITE .

إذا تغيرت الجملة 150 إلى :

```
150 RESTORE$
```

بذلك يعاد المؤشر الحرفى دون المؤشر الرقى . ولذلك يتم تحديد C1 و C2 بالقيم 6 و 8 ولكن المتغيرين F1$ و F2$ سيتم تحديد قيمهما بالحروف RED و GREEN على الترتيب .

والآن نفرض أن الجملة 150 تغيرت إلى :

```
150 RESTORE
```

بذلك يعاد كل من المؤشرين وتكون نتيجة ذلك : C1 = 2 و C2 = 4 و F1$=RED و F2$=GREEN . وأخيراً ، إذا ألغيت الجملة رقم 150 بالكامل ، فإن نتيجة الجملة 160 تكون C1 = 6 و C2 = 8 و F1$=BLUE و F2$=WHITE

يجب أن يتضح أن القيم التى تحدد بها A و B و M$ و N$ لاتتأثر بجمل RESTORE و READ التالية فى هذا المثال .

٥ - ٧ ملاحظـــات ختـــامية CLOSING REMARKS

إننا بهذا الفصل ننهى مناقشتنا لخصائص « الأساسية » للغة البيسك ولقد رأينا أن استخدام الدوال المكتبية يمكن المبرمج من القيام بتنفيذ عمليات معينة (مثل بتر الجزء العشرى ، تباعد بيانات المخرجات ، ... الخ) وذلك بجانب العمليات الرياضية الشائعة . يمكن تخزين والتعامل مع مجموعات من الأرقام والحروف وذلك باستخدام القوائم والجداول . ويمكن لكميات كبيرة من البيانات أن تخزن بطريقة ملائمة وتعطى لمتغيرات فى البرنامج أو لعناصر مجموعة متراصة وذلك من خلال استخدام جملتى READ و DATA . يترتب على هذه الخصائص تيسير برمجة كثير من المسائل المتنوعة .

اسئلة للمراجعة

Review Questions

٥ – ١ ماهى الدوال المكتبية ؟ وما هو الغرض الأساسى من استخدامها ؟
٥ – ٢ ماهى الأسماء الأخرى التى تطلق أحياناً على الدوال المكتبية ؟
٥ – ٣ اذكر عدة أسماء من الدوال المكتبية الاكثر شيوعاً ؟ ثم اذكر الغرض من كل من هذه الدوال المكتبية ؟
٥ – ٤ ماهو المقصود من الخلاصة ؟ وهل تتطلب كل الدوال المكتبية خلاصات ؟
٥ – ٥ كيف تستخدم الدوال المكتبية فى برنامج بيسك ؟
٥ – ٦ ماذا يحدث لو أعطيت دالة مكتبية قيمة سالبة ولكنها تتطلب خلاصة موجبة ؟
٥ – ٧ ماهو المقصود من البتر ؟ اذكر مثالا يوضح البتر .
٥ – ٨ ماهو الغرض من دالة INT ؟ ماذا يحدث عندما تستقبل دالة INT خلاصة موجبة ؟ وعندما تستقبل خلاصة سالبة ؟
٥ – ٩ ماهو الغرض من دالة TAB ؟ وفى أى جملة تستعمل ؟
٥ – ١٠ هل يمكن استخدام صيغة رياضية كخلاصة لدالة مكتبية ؟ وهل يمكن الرجوع لدالة مكتبية أخرى لهذا الغرض ؟
٥ – ١١ ماذا يحدث لو أعطينا قيمة غير صحيحة كخلاصة لدالة مكتبية مع العلم أنها تتطلب خلاصة صحيحة ؟
٥ – ١٢ ماهو المقصود من القائمة ؟ الجدول ؟
٥ – ١٣ ماهو المقصود من مجموعة متراصة ذات بعد واحد ؟ مجموعة متراصة ذات بعدين ؟ قارن إجابتك بإجابة السؤال السابق .
٥ – ١٤ ماهو المقصود بعناصر القائمة أو الجدول ؟ وماذا تمثل هذه العناصر ؟
٥ – ١٥ هل يمكن أن تحتوى مجموعة متراصة واحدة أرقاماً وحروفاً ؟
٥ – ١٦ ماهو المتغير ذا الدليل ؟ كيف نشير إلى متغير ذى دليل معين ؟
٥ – ١٧ هل يمكن استخدام الصيغة الرياضية كدليل ؟ وهل يمكن الإشارة إلى دالة مكتبية واستخدامها لنفس الغرض ؟
٥ – ١٨ ماهى القيود التى تنطبق على القيم التى يمكن أن تأخذها الأدلة ؟
٥ – ١٩ ماهو الغرض من جملة DIM ؟ ومتى يجب أن تظهر هذه الجملة فى برنامج بيسك ؟
٥ – ٢٠ لخص قواعد كتابة جملة DIM .
٥ – ٢١ هل يمكن لصيغة رياضية أو مرجع لدالة مكتبية أن تظهر فى جملة DIM ؟
٥ – ٢٢ هل يوجد غرض ما فى توصيف حجم مجموعة متراصة صغيرة فى جملة DIM ؟ اشرح .
٥ – ٢٣ ماهو الغرض من جملتى READ و DATA ؟
٥ – ٢٤ لخص قواعد كتابة جملة READ .
٥ – ٢٥ لخص قواعد كتابة جملة DATA .
٥ – ٢٦ هل تتطلب كل جملة READ جملة بيانات خاصة بها ؟ إشرح .
٥ – ٢٧ ماهو المقصود من كتلة البيانات ؟ ومما تتكون كتلة البيانات ؟
٥ – ٢٨ هل البيانات التى تدخل من خلال جملة INPUT دائمة كالبيانات التى تدخل من خلال جملتى READ-DATA ؟ إشرح .
٥ – ٢٩ لخص القواعد التى يجب أن تلاحظ عند وضع عناصر البيانات فى كتلة البيانات .
٥ – ٣٠ أين توضع جمل DATA غالباً فى برنامج بيسك ؟ ولماذا ؟
٥ – ٣١ كيف يمكن القيام بعمليات الإدخال / الإخراج لقائمة أو جدول ؟
٥ – ٣٢ ناقش الغرض من المؤشرات كذا ناقش استخدامها مقترنة بكتلة البيانات .
٥ – ٣٣ ماهو الغرض من جملة RESTORE ؟ اذكر ثلاث طرق مختلفة يمكن أن تكتب بها هذه الجملة .

مسائل محلولة
Solved Problems

٥ – ٣٤ أكتب جملة بيسك تناظر كلا من المعادلات الجبرية التالية :

(أ) $z = \tan t$

```
10 LET Z=TAN(T)
```

(ب) $w = \log_e(v)$

```
10 LET W=LOG(V)
```

(ح) $y = ae^{bx}\sin cx$

```
10 LET Y=A*EXP(B*X)*SIN(C*X)
```

(د) $x_1 = \dfrac{-b + \sqrt{b^2 - 4ac}}{2a}$

```
10 LET X1=(-B+SQR(B↑2-4*A*C))/(2*A)
```

٥ – ٣٥ أكتب جملة بيسك لكل من المواقف التالية :

(أ) قرر القيمة المطلقة للفرق بين المتغيرين U و V ثم حدد قيمة المتغير W بالناتج .

```
10 LET W=ABS(U-V)
```

(ب) قرر إشارة x . فإذا كانت x سالبة تفرع للجملة 50 ، وإذا كانت x مساوية للصفر تفرع للجملة 20 ، وإذا كانت x موجبة تفرع للجملة 170 .

```
100 ON SGN(X)+2 GO TO 50,20,170
```

(ح) قرر أكبر رقم صحيح لايتعدى قيمة z جبرياً ، حيث $z = x^2 - y^2$ ثم اعط هذا الرقم الصحيح للمتغير I .

```
10 LET I=INT(X↑2-Y↑2)
```

(د) فى (ح) عالية ، إذا كانت $x = 2.5$ و $y = 6.3$ ماهى القيمة التى تعطى للمتغير I ؟

$x^2 - y^2 = -33.44$ ومن ثم $I = -34$

(ه) اطبع القيم X$ و X و Y$ و Y على نفس السطر . اجعل السلسلة الممثلة بالمتغير X$ تبدأ من العمود رقم 10 ، ويتبعها فوراً قيمة X . وبالمثل اجعل Y$ تبدأ من العمود رقم 46 ، ويتبعها فوراً قيمة Y .

```
100 PRINT TAB(9);X$;X;TAB(45);Y$;Y
```

٥ – ٣٦ يبين كل مثال مما يأتى مرجعاً إلى متغير أو أكثر من المتغيرات ذات الأدلة . صف نوع المجموعة المتراصة التى يشار إليها فى كل حالة :

(أ)

```
10 DIM C1(100),N$(100,3)
```

C1 قائمة عددية ، N$ جدول حرفى .

(ب)

```
50 LET P(I)=P(I)+Q(I,J)
```

P قائمة عددية ، Q جدول رقمى

(حـ) 100 IF A$(5)=G$ THEN 220

A$ قائمة حرفية

(د) . 150 ON X(K(I),J(I)) GO TO 100,20,180,20,250

X جدول رقمى و K و J قوائم رقية .

(هـ) 200 PRINT X$(K),N1(K),N2(K)

X$ قائمة حرفية ، N1 و N2 قوائم رقية

٣٧ – اكتب جملة أو أكثر للقيام بكل من العمليات الآتية :

(أ) اجمع العناصر ال 100 الأولى من القائمة الرقية T .

```
10 LET S=0
20 FOR I=0 TO 99
      LET S=S+T(I)
40 NEXT I
```

(ب) اطبع الأرقام الزوجية من القائمة الرقية T لقيم الدليل التى تتراوح مابين : 0 و 100 أى اطبع T(0) و T(2) و T(4) و............ T(100) .

```
10 FOR I=0 TO 100 STEP 2
20      PRINT T(I);
30 NEXT I
```

(حـ) احسب مجموع كل عناصر الجدول الرقى P . اجعل M تشير إلى عدد الصفوف و N تشير إلى عدد الأعمدة .

```
10 LET S=0
20 FOR I=1 TO M
30     FOR J=1 TO N
40         LET S=S+P(I,J)
50     NEXT J
60 NEXT I
```

(د) اطبع عناصر العمود الثالث من الجدول الحرفى K$. اعرض المخرجات فى صورة عمود إبتداء من العمود رقم 12 (يشير رقم العمود هنا إلى موضع الطباعة على الورق) .

```
10 FOR I=1 TO M
20      PRINT TAB(12);K$(I,3)
30 NEXT I
```

٣٨ – اكتب جمل READ و DATA مناسبة لكل موقف موصوف فيما يلى :

(أ) حدد القيم 6– 1.6 E و 500– و 4077. و MAY و OCTOBER و 100 و 110 و 120 و 130 و 140 و 150 للمتغيرات C1 و C2 و C3 و X$ و Y$ و Z(1) و Z(2) و Z(3) و Z(4) و Z(5) و Z(6) . اطبع كل متغير ذى دليل منفصلا فى جملة READ .

```
10 READ C1,C2,C3,X$,Y$,Z(1),Z(2),Z(3),Z(4),Z(5),Z(6)
   ...
200 DATA -1.6E-6,-500,.4077,MAY,OCTOBER,100,110,120,130,140,150
```

يمكن أيضاً تقسيم جمل READ و DATA عند الرغبة ، مثلا ،

```
 10 READ C1,C2,C3,X$,Y$
 20 READ Z(1),Z(2),Z(3),Z(4),Z(5),Z(6)
      ...
200 DATA -1.6E-6,-500,.4077
210 DATA MAY,OCTOBER,100,110
220 DATA 120,130,140,150
```

(ب) حدد القيم المعطاة فى الجزء (أ) عالية لمتغيراتها بالترتيب . استخدم حلقة FOR-TO لعناصر المجموعة المتراصة .

```
 10 READ C1,C2,C3,X$,Y$
 20 FOR I=1 TO 6
 30     READ Z(I)
 40 NEXT 1
     ...
200 DATA -1.6E-6,-500,.4077,MAY,OCTOBER
210 DATA 100,110,120,130,140,150
```

(حـ) حدد القيم المعطاة فى الجزء (أ) عالية لمتغيراتها بالترتيب كما فى جزء (ب) . وعند جزء لاحق فى البرنامج أعد مؤشر السلاسل الحرفية وحدد القيم الحرفية MAY و OCTOBER للمتغيرين F$ و G$.

```
 10 READ C1,C2,C3,X$,Y$
 20 FOR I=1 TO 6
 30     READ Z(I)
 40 NEXT I
     ...
100 RESTORE$
110 READ F$,G$
     ...
200 DATA -1.6E-6,-500,.4077,MAY,OCTOBER
210 DATA 100,110,120,130,140,150
```

مسائل تكميلية

Supplementary Problems

٥ - ٣٩ أوجد الدوال المكتبية المتاحة على جهازك . (أشر إلى مرجع البيسك الذى تم نشره بواسطة صانع الحاسب الخاص بك) صف غرض كل دالة مكتبية وقرر تماماً كيف يرجع إلى كل دالة .

٥ - ٤٠ اكتب جملة بيسك تناظر كلا من المعادلات الجبرية التالية :

(أ) $y = \sqrt{\sin x - \cos x}$

(ب) $p = qe^{-at}$

(حـ) $c = \log_e \sqrt{|a+b|} + \log_e \sqrt{|a-b|}$

(د) $w = ||u-v| - |u+v||$

(هـ) $z = \cos(x + \arctan y)$

٥ - ٤١ اكتب جملة بيسك لكل من المواقف التالية :

(أ) قرر ماهى إشارة الكمية $(ab - cd)/(f+g)$ إقفز إلى الجملة 75 إذا كانت الكمية موجبة ، وإلى الجملة 260 إذا كانت الكمية مساوية لصفر ، وإلى الجملة 135 إذا كانت الكمية سالبة .

(ب) اطبع التالى على سطر واحد من النهاية الطرفية المركزية " X= " تتبعها قيمة المتغير X ، ثم " Y= " وتتبعها قيمة المتغير Y ، ثم " Z = " وتتبعها قيمة المتغير Z . ابدأ الطباعة عند الأعمدة 4 و28 و 52 على الترتيب .

(حـ) قرر ما إذا كانت قيمة N زوجية أو فردية ، مع فرض أن N لها قيم صحيحة موجبة . (**تلميح** : قارن قيمةN/2 بالقيمة N/2 بعد بتر الكسر من الرقم) .

(د) فى الجزء (حـ) عالية ، ماذا يحدث إذا كانت قيمة N قيمة صحيحة سالبة ؟

٥ — ٤٢ يبين كل مثال مما يلى الإشارة إلى متغير أو أكثر من المتغيرات ذات الأدلة . صف نوع المجموعة المتراصة المشار إليها فى كل حالة .

(أ) `75 LET N$(3)="ERROR-CHECK"`

(ب) `20 DIM A(12,25),A$(12,25),B(12),C$(25)`

(حـ) `100 PRINT P$(I),P(I,J)`

(د) `50 IF Z(J1,J2)<10 THEN 185`

٥ — ٤٣ مبين فيما يلى عدة جمل بيسك وجمل متسلسلة تحتوى على متغيرات ذات أدلة . بعض الأمثلة مكتوبة بصورة غير صحيحة . تعرف على كل الأخطاء .

(أ) `50 LET C(I,J)=(3*X↑..−2*Y↑3)/(17*Z)`

(ب) `75 LET F(K,5)=Q(K+1,J)+R(K,J+1)`

(جـ)
```
20 INPUT M,N
30 DIM A(M,N),X(N),Y(M)
```

(د) `10 DIM K(100),W(10,20),C1,C2,K(100)`

(هـ) `150 LET S=S+T(K,−3)`

(و) `200 LET X(K(I))=Y(K(I+1))+Z(K(I−1))`

٥ — ٤٤ اكتب جملة أو أكثر للقيام بكل من العمليات التالية :

(أ) احسب الجذر التربيعى لمجموع مربعات الأعداد ، للعناصر الفردية الأولى من القائمة الرقية والتى عددها 100 ، أى ، احسب $[X(1)^2+X(3)^2+X(5)^2+\cdots+X(199)^2]^{1/2}$

(ب) احسب عناصر الجدول الرقى H والذى يحتوى على 8 صفوف و 12 عموداً . تقرر قيمة كل عنصر من الجدول H بالصيغة الرياضية :

$$h_{ij} = \frac{1}{i+j-1}$$

(حـ) تحتوى قائمة رقية K على عدد N من العناصر . اطبع قيمة كل دليل والعنصر المناظر للعناصر التى لاتتعدى قيمتها 15 . أعرض المخرجات فى عمودين بقيمة الدليل فى العمود الأول والمتغير ذى الدليل المناظر فى العمود الثانى . اعط عنواناً لكل عمود . أبدأ العمود الأول من المخرجات فى الموضع رقم 8 (يشير رقم الموضع هنا إلى مكان الطباعة على الورق) ، والعمود الثانى من المخرجات فى الموضع رقم 44 .

(د) جدول رقى W له عدد K من الصفوف و K من الأعمدة احسب حاصل ضرب حدود القطر الرئيسى لجدول W ، حيث يتغير القطر الرئيسى من اليسار الأعلى إلى اليمين الأسفل ، أى ، احسب :

$$W(1,1)*W(2,2)*W(3,3)*\ldots*W(K,K).$$

(هـ) اطبع عناصر العمود الرابع من الجدول الحرفى M$ إعرض المخرجات فى صورة عمود ، مبتدئاً فى الموضع رقم 10 (يشير رقم الموضع هنا إلى مكان الطباعة على الورق) . إفرض أن M$ تحتوى على عدد M من الصفوف .

(و) كرر المسألة (هـ) عاليه مع عرض المخرجات فى شكل صف مع ترك مسافة واحدة بين كل عنصر .

(ز) إطبع العناصر فى الصف الخامس من الجدول الحرفى M$ إعرض المخرجات فى شكل صف مع ترك مسافة واحدة بين كل عنصر . إفرض أن M$ تحتوى على عدد N من الأعمدة .

٥ – ٤٥ اكتب جمل READ و DATA تلائم كل موقف موصوف أدناه :

(أ) تحدد القيم التالية للقائمة L$ والمتغيرات P و Q و R و H$ والجدول الرقمى T .

L$(1)=WHITE	P=2.25E+5	T(1,1)=1	T(2,1)=−2
L$(2)=YELLOW	Q=6.08E−9	T(1,2)=−3	T(2,2)=4
L$(3)=ORANGE	R=−1.29E+12	T(1,3)=5	T(2,3)=−6
L$(4)=RED	H$=RESTART	T(1,4)=−7	T(2,4)=8

اطبع قائمة بالمتغيرات ذات الأدلة منفصلة فى جمل READ .

(ب) كرر الجزء (أ) عاليه مستخدماً الحلقات FOR-TO لعناصر المجموعة المتراصة .

(حـ) كرر الجزء (أ) عاليه ، مستخدماً الحلقات FOR-TO لعناصر المجموعة المتراصة ، كما فى الجزء (ب) . وعند جزء لاحق من البرنامج أعد المؤشر الرقمى ثم حدد القيم 2.25 E + 5 و 6.08E — 9 و 12 + 1.29 E— إلى المتغيرات P1 و Q1 و R1 .

(د) كرر الجزء (ا) عاليه ، مستخدما حلقات FOR-TO لعناصر المجموعة المتراصة كما فى شكل (ب) . وعند جزء لاحق من البرنامج ، أعد المؤشر ليحدد القيم الحرفية WHITE و YELLOW و ORANGE و RED للمتغيرات A1$ و A2$ و A3$ و A4$.

مسائل للبرمجة

Programming Problems

٥ – ٤٦ غير البرنامج المبين فى مثال ٥ – ٥ بحيث لايحسب اللوغاريتم عندما تكون قيمة x مساوية للصفر . ضمن البرنامج احتياطى لطباعة 8 نجوم متتالية للوغاريتم صفر . وبذلك يشير إلى حالة طفح .

٥ – ٤٧ اكتب برنامج بيسك مماثل للمثال ٥ – ٥ لتولد جدول من x و $\sin^2 x$ و $\cos^2 x$ و $\tan^2 x$ و $\cot^2 x$ و $\sec^2 x$ و $\mathrm{cosec}^2 x$ (لاحظ أن $\sec x = 1/\cos x$ و $\mathrm{cosec}\, x = 1/\sin x$) أنشأ عدد 101 من المداخل لقيم متباعدة بالتساوى حيث تتراوح قيم x ما بين صفر و π (وهذا يعنى : اجعل قيمة x مساوية 0 و $\pi/100$ و $2\pi/100$ و ... و $99\pi/100$ و π) تأكد من أن كل المخرجات لها عناوين ملائمة .

٥ – ٤٨ اكتب برنامج بيسك ينشىء جدولا من القيم للمعادلة :

$$y = 2e^{-0.1t} \sin 0.5\, t$$

حيث تتغير t مابين 0 و 60 اسمح لحجم الزيادة فى t أن تدخل كعامل إدخال .

٥ — ٤٩ وسع برنامج الكلمات غير الممتزجة فى مثال ٥ — ٩ بحيث يطبع كل التوافقيات الممكنة من حرفين أو ثلاثة حروف أو أربعة حروف وذلك لأى أربعة حروف معطاة . اختر مجموعة من أربعة حروف ثم نفذ البرنامج . تعرف على كل الكلمات الإنجليزية الصالحة وذلك بفحص المخرجات مرئياً . هل يمكنك عمل ذلك بدون تشغيل البرنامج . كم عدد التوافقيات المختلفة التى يمكن توليدها من حرفين أو أكثر ؟

٥ — ٥٠ وسع البرنامج فى المثال ٥ — ١٤ والذى يمكن به إعادة تنظيم قائمة من الأرقام بأى طريقة من الطرق الأربع التالية :

(أ) من الأصغر إلى الأكبر جبرياً (من أكبر قيمة سالبة إلى أكبر قيمة موجبة) .

(ب) من الأصغر إلى الأكبر فى القيمة (وذلك بإغفال الإشارات) .

(جـ) من الأكبر إلى الأصغر جبرياً .

(د) من الأكبر إلى الأصغر فى القيمة .

(لاحظ أنه ليس من المهم أن تكون عناصر القائمة بها قيما موجبة) .

إكتب البرنامج بطريقة تمكنه من القيام بعمل إعادة تنظيم واحدة فقط فى كل مرة ينفذ فيها البرنامج . ضمن فى البرنامج متغيراً يمكن إدخال قيمته من خلال جملة INPUT فى كل مرة ينفذ فيها البرنامج . اجعل إعادة تنظيم القائمة من الأرقام يمكن تقريرها بواسطة القيمة التى تعطى لهذا المتغير (مثال ، إذا كانت A = 1 فإن إعادة التنظيم ستكون من الأصغر للأكبر جبريا ، وإذا كانت A = 2 فإن اعادة التنظيم ستكون من الاصغر للأكبر فى القيمة ، . . . الخ)

استخدم البرنامج لإعادة تنظيم الأرقام المعطاة فى الجدول ٥ — ٣ . أعد تنظيم الأرقام بالطرق الأربع .

جدول ٥ — ٣

43	−85	−4	65
−83	10	−71	−59
61	−51	−45	−32
14	49	19	23
−94	−34	−50	86

٥ — ٥١ اكتب برنامج بيسك يعيد تنظيم قائمة من الكلمات بترتيب هجائى . ولعمل ذلك ، أدخل الكلمات فى قائمة ، حيث يمثل كل عنصر كلمة واحدة . بعد ذلك يمكن ترتيب قائمة الحروف أبجدياً بنفس الطريقة التى يمكن بها إعادة تنظيم قائمة من الأرقام من الأصغر إلى الأكبر (أنظر مثال ٥ — ١٤) .

استخدم البرنامج ليعيد تنظيم الأسماء المعطاة فى جدول ٥ — ٤ . كن حريصاً فى التعامل مع الحروف الأولى .

جدول ٥ — ٤

Washington	Arthur
Adams, J.	Cleveland
Jefferson	Harrison, B.
Madison	McKinley
Monroe	Roosevelt, T.
Adams, J. Q.	Taft
Jackson	Wilson
Van Buren	Harding
Harrison, W. H.	Coolidge
Tyler	Hoover
Polk	Roosevelt, F. D.
Taylor	Truman
Fillmore	Eisenhower
Pierce	Kennedy
Buchanan	Johnson, L. B.
Lincoln	Nixon
Johnson, A.	Ford
Grant	Carter
Hayes	Reagan
Garfield	

٥ – ٥٢ أعد كتابة برنامج البيسك فى مثال ٥ – ١٥ بحيث يحسب حاصل ضرب العناصر فى كل صف وفى كل عمود .

٥ – ٥٣ اكتب برنامج البيسك يولد جدولا من معاملات الأرباح المركبة ، F/P حيث :

$$F/P = [1 + (i/100)]^n$$

وتمثل i فى هذه الصيغة الرياضية نسبة الربح السنوى معبراً عنها كنسبة مئوية ، وتمثل n عدد السنوات .

اجعل كل صف فى الجدول يناظر قيمة مختلفة من n ، وذلك لتتراوح n من 1 إلى 30 (من ثم 30 صفاً) .
واجعل كل عمود يمثل نسبة ربح مختلفة ضمن نسب الأرباح التالية :

4 و 4.5 و 5 و 5.5 و 6 و 6.5 و 7 و 7.5 و 8 و 8.5 و9 و 9.5 و 10 و 11 و 12 و 15 فى المائة (من ثم 16 عموداً) . تأكد من إعطاء عناوين مناسبة لكل من الأعمدة والصفوف .

٥ – ٥٤ اكتب برنامج بيسك (BASIC) يقرأ مجموعة من درجات الحرارة ، ثم يحسب متوسطا لها ثم بعد ذلك يحسب انحراف كل درجة حرارة عن المتوسط .

يعرف الانحراف كالتالى : $D = T(I) - A$

حيث تمثل A متوسط درجة الحرارة . لاحظ أن الانحراف سيكون موجباً إذا كانت درجة الحرارة أعلى من المتوسط ، وسوف يكون سالباً إذا كانت درجة الحرارة أقل من المتوسط .

اطبع متوسط درجة الحرارة ، تتبعها ثلاثة أعمدة تحتوى على القيم I و $T(I)$ و D على الترتيب . وتأكد من أن كل شىء له عنوان واضح .

اختبر البرنامج باستخدام مجموعة درجات الحرارة التالية : 28.2 و 29.3 و 33.7 و 42.0 و 58.4 و 71.3 و 84.1 و 83.8 و74.5 و53.9 و 41.6 و 34.4

٥ – ٥٥ وسع برنامج حساب متوسطات درجات طالب (المسألة ٤ – ٤٨ (i)) حيث يمكن حساب انحراف متوسط كل طالب عن المتوسط العام للفصل . اطبع متوسط الفصل ، يتبعه إسم الطالب ، ثم درجات الامتحان ، والدرجة النهائية والانحراف عن متوسط الفصل . تأكد أنه تم تنظيم المخرجات منطقياً وتم إعطاؤها عناوين واضحة .

٥ – ٥٦ إدرس القائمة التالية لبعض الدول وعاصمتها .

Canada	Ottawa
England	London
France	Paris
India	New Delhi
Israel	Jerusalem
Italy	Rome
Japan	Tokyo
Mexico	Mexico City
People's Republic of China	Peking
United States	Washington
U.S.S.R.	Moscow
West Germany	Bonn

اكتب برنامج بيسك من النوع التخاطبى والذى سوف يقبل إسم الدولة كمدخل ويطبع العاصمة المناظرة لها والعكس صحيح .

٥ – ٥٧ عدة مسائل فنية من أنواع مختلفة تم توصيفها أدناه . جهز مخططاً تمهيدياً مفصلا وخريطة سير عمليات مناظرة ثم برنامج بيسك كامل لكل من هذه المسائل .

(أ) نفرض أن لدينا جدولا رقمياً A له M صف و N عمود ولدينا أيضاً قائمة رقمية لها N عنصر . والمطلوب توليد قائمة رقمية Y وذلك بالقيام بالعمليات التالية .

```
Y(1)=A(1,1)*X(1)+A(1,2)*X(2)+···+A(1,N)*X(N)
Y(2)=A(2,1)*X(1)+A(2,2)*X(2)+···+A(2,N)*X(N)
...
Y(M)=A(M,1)*X(1)+A(M,2)*X(2)+···+A(M,N)*X(N)
```

اطبع المدخلات (أى قيم عناصر الجدول A والقائمة X) ، تتبعها القيم المحسوبة لعناصر القائمة Y .

استخدم البرنامج لمعالجة مجموعة البيانات التالية :

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\ 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 \\ 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 \\ 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 \end{bmatrix}$$

$$X = \begin{bmatrix} 1 \\ -8 \\ 3 \\ -6 \\ 5 \\ -4 \\ 7 \\ -2 \end{bmatrix}$$

(ب) نفرض أن A جدول رقمى له K صف و M عمود ، وأن B جدول رقمى آخر له M صف و N عمود والمطلوب حساب عناصر جدول رقمى C حيث يمكن تحديد كل عنصر فى الجدول بواسطة :

```
C(I,J)=A(I,1)*B(1,J)+A(I,2)*B(2,J)+···+A(I, M)*B(M, J)
```

وذلك لقيم I تتراوح من 1 و 2 و... إلى K ولقيم J تتراوح من 1 و 2 و.... إلى N اطبع عناصر الجداول الرقمية A و B و C .

استخدم البرنامج لمعالجة مجموعة البيانات التالية :

$$A = \begin{bmatrix} 2 & -1/3 & 0 & 2/3 & 4 \\ 1/2 & 3/2 & 4 & -2 & 1 \\ 0 & 3 & -9/7 & 6/7 & 4/3 \end{bmatrix}$$

$$B = \begin{bmatrix} 6/5 & 0 & -2 & 1/3 \\ 5 & 7/2 & 3/4 & -3/2 \\ 0 & -1 & 1 & 0 \\ 9/2 & 3/7 & -3 & 3 \\ 4 & -1/2 & 0 & 3/4 \end{bmatrix}$$

(ج) يمكن حساب متعددة الحدود (لجندر) *legendre* بواسطة الصيغ الرياضية التالية :

$$P_0 = 1$$
$$P_1 = x$$
$$\cdots$$
$$P_n = \left(\frac{2n-1}{n}\right) x P_{n-1} - \left(\frac{n-1}{n}\right) P_{n-2}$$

حيث تأخذ n القيم 2 و3 و4 ,...... وأن x أى رقم بين 1 - و 1 +

اكتب برنامج بيسك يولد جدول للقيم P_n مقابل قيم x لأى قيمة n من 1 إلى 10 . ولد 201 قيمة من P_n فى كل جدول مبنية على أساس قيم متساوية التباعد للمتغير x (أى اجعل x تأخذ القيم 1.00— و0.99— و0.98— و . . 0.01— و0 و 0.01 و .... و 0.98 و 0.99 و 1.00) واستخرج النتائج فى صورة عمود واضح ومقروء .

(د) إدرس تسلسلا لأرقام حقيقية x_i حيث تأخذ i القيم 1 و 2 ، .... و M . يعرف المتوسط كالتالى :

$$\bar{x} = \frac{x_1 + x_2 + \cdots + x_M}{M}$$

الانحراف عن المتوسط هو

$$d_i = (x_i - \bar{x})$$

ومعيار الانحراف هو

$$\sigma = \left[\frac{d_1^2 + d_2^2 + \cdots + d_M^2}{M}\right]^{1/2}$$

اقرأ أول M عنصر من مجموعة متراصة لها بعد واحد . ثم احسب مجموع هذه العناصر ، والانحرافات ومعيار الانحراف ، والقيمة القصوى الجبرية والقيمة الصغرى الجبرية . طبق هذا البرنامج على بيانات درجات الحرارة المعطاة فى المسألة ٥ - ٤٥

كرر العمل لعدد K من المجموعات المتراصة المختلفة . ثم احسب المتوسط العام ومعيار الانحراف العام والقيمة المطلقة للحد الأقصى (الأكبر) والقيمة المطلقة للحد الأدنى (الأصغر جبريا) .

(هـ) اكتب برنامج بيسك لحساب التفاوت ٮ ، لقائمة من الأرقام بطريقتين باستخدام الصيغ الرياضية :

$$\bar{v} = \frac{1}{M}[(x_1 - \bar{x})^2 + (x_2 - \bar{x})^2 + \cdots + (x_M - \bar{x})^2]$$

و

$$\bar{v} = \frac{1}{M}(x_1^2 + x_2^2 + \cdots + x_M^2) - \bar{x}^2$$

وفى هذه الصيغ الرياضية تحسب قيمة المتوسط $\bar{x}$ كما فى الصيغة الرياضية :

$$\bar{x} = \frac{1}{M}(x_1 + x_2 + \cdots + x_M)$$

حيث M هى عدد القيم فى القائمة .

رياضياً ، يمكن رؤية أن الصيغتين الرياضيتين لحساب ٮ متماثلتان عندما تكون الأرقام المعطاة لها قيم متقاربة جداً من بعضها ، ومن ثم فان القيمة التى نحصل عليها ٮ من استخدام الصيغة الرياضية الثانية اعتبارياً خطأ . والسبب فى ذلك أننا يجب أن نحسب الفرق بين قيمتين تقريباً متساويتين . مثل هذه الفروق المحسوبة يمكن أن تكون عدم دقتها عالية . وتنتج الصيغة الرياضية الأولى نتائج أكثر دقة تحت هذه الظروف .

وضح أن الجملة السابقة صحيحة وذلك بحساب اختلاف البيانات المعطاة فى جدول ٥-٥ (القيمة الصحيحة هى : ($\bar{v}$ = 0.00339966

جدول ٥ — ٥

99.944	100.054	100.059	100.061
100.039	100.066	100.029	100.098
99.960	99.936	100.085	100.038
100.093	99.932	100.079	100.024
99.993	99.913	100.095	100.046

لاحظ : أن هذا المثال يبرهن على عدم صحة الاعتقاد الشائع والخاطئ أن الحاسب دائماً ما ينتج إجابات صحيحة جداً) .

(و) تحسب تكاليف رهن منزل بطريقة يفترض فيها أن يدفع مبلغ ثابت شهرياً خلال مدة الرهن للهيئة المقرضة . إلا أن الجزء من إجمالى المدفوعات الشهرية الذى يمثل الفوائد على الرصيد غير المدفوع من القرض يختلف من شهر لآخر . خلال مدة الرهن الأولى . تمثل معظم المدفوعات الشهرية المبالغ المطلوبة لتسديد الفوائد على الديون وكسر صغير فقط من المدفوعات الشهرية يمثل تسديدات تدريجية للقرض ، وبذا يتناقص الرصيد غير المدفوع تدريجياً ، أى يترتب على ذلك أن تتناقص مدفوعات الفوائد الشهرية بينما يزداد المبلغ الموجه لتسديد الدين الأصلى ، وبذلك فان رصيد القرض يخفض بنسبة متصاعدة .

يعلم تماماً المزمع على شراء منزل المبلغ الذى يطلب اقتراضه والوقت اللازم لتسديد الديون . ثم يسأل الهيئة المقرضة بعد ذلك عن الأقساط الشهرية حسب سعر الفائدة السائد . ويجب أن يهتم أيضاً بمقدار الفوائد التى تخصم من المدفوعات الشهرية ، وإجمالى الفوائد التى تم دفعها من أول القرض ، والمبالغ المتبقية عليه من أصل الدين والواجب دفعها للهيئة المقرضة عند نهاية كل شهر .

اكتب برنامج بيسك يمكن أن يستخدم بواسطة الهيئة المقرضة يمد العميل بكل هذه المعلومات . نفرض أنه سبق توصيف قيمة القرض ، نسبة الربح السنوية ومدة القرض وأن المدفوعات الشهرية تحسب بواسطة الصيغة الرياضية :

$$A = iP\left[\frac{(1+i)^n}{(1+i)^n - 1}\right]$$

حيث A هى المدفوعات الشهرية بالدولار .

P إجمالى القرض بالدولار .

i نسبة الربح الشهرى معبرا عنه برقم كسرى ، (مثال ½% يجب كتابتها 0.005) .

n عدد الأقساط الشهرية .

ويمكن حساب الفائدة الشهرى المدفوعة من الصيغة الرياضية .

$$I = iB$$

حيث I فوائد المدفوعات الشهرية بالدولار .

B الرصيد غير المدفوع بالدولار .

ويساوى الرصيد غير المدفوع ببساطة القرض الأصلى مطروحاً منه المدفوعات السابقة لتسديد الدين الأصلى . المدفوعات الشهرية لتسديد الدين الأصلى ، أى الكية المستخدمة لتخفيض الرصيد غير المدفوع ، هى ببساطة

$$T = A - I$$

حيث T هى المدفوعات الشهرية لتسديد الدين الأصلى .

استخدم البرنامج لحساب تكاليف قرض مقداره 30,000$ ولمدة 25 سنة وبنسبة فوائد سنوية 8% ثم بعد ذلك كرر الحسابات لنسبة فائدة سنوية 8.5% . ما هو الأثر الفعلي على قيمة القرض نتيجة زيادة الفائدة بمقدار 0.5% طوال مدة الرهن ؟

(ز) تعرف الطريقة المستخدمة لحساب تكلفة رهن منزل فى المسألة ٥ – ٥٧ سابقاً بطريقة المدفوعات الثابتة ، حيث أن أقساط المدفوعات الشهرية ثابتة . وإذا فرضنا بدلا من ذلك أن المدفوعات الشهرية تم حسابها بطريقة الربح البسيط . أى بفرض أن تسديد القرض يتم بأقساط شهرية ثابتة قيمتها :

$$T = P/n$$

وبالإضافة إلى ذلك ، تسدد الفائدة على الرصيد غير المسدد بمقدار ثابت شهرياً فان :

$$I = iB$$

وبذلك يكون المبلغ المسدد شهرياً (A) هو حاصل جمع T و I أى $A = T + I$ ويتناقص كل شهر كلما تلاشى الرصيد غير المسدد .

اكتب برنامج بيسك لحساب تكلفة رهن منزل باستخدام هذه الطريقة فى السداد . عنون المخرجات بوضوح . واستخدم البرنامج لحساب تكلفة رهن مقداره 30,000 $ لمدة 25 سنة بنسبة فائدة 8% سنوياً . قارن هذه النتائج بالنتائج التى تم الحصول عليها من المسألة ٥ – ٥٧ (و) .

(ح) نفرض أننا أعطينا مجموعة من القيم المجدولة لكل من y مقابل x أى :

y_0	y_1	y_2	$\cdots$	y_n
x_0	x_1	x_2	$\cdots$	x_n

ونرغب فى الحصول على قيمة y عند أى قيمة x تقع ما بين قيمتين فى الجدول . والطريقة الشائعة لحل هذه المسألة هى طريقة **الاستكمال** أى بتمرير كثيرة الحدود $y(x)$ خلال n نقطة حيث :
$y(x_0) = y_0, y(x_1) = y_1, \ldots\ldots\ldots\ldots, y(x_n) = y_n$ وبعد ذلك حساب y عند النقطة المطلوبة للمتغير x .

والطريقة الشائعة للقيام بهذا الاستكمال هى استخدام **صيغة لاجرانج** *lagrange* لاستكمال كثيرة الحدود . ولعمل ذلك نكتب :

$$y(x) = f_0(x) \cdot y_0 + f_1(x) \cdot y_1 + \cdots + f_n(x) \cdot y_n$$

حيث أن $f_i(x)$ كثيرة الحدود بحيث

$$f_i(x) = \frac{(x - x_0)(x - x_1)\cdots(x - x_{i-1})(x - x_{i+1})\cdots(x - x_n)}{(x_i - x_0)(x_i - x_1)\cdots(x_i - x_{i-1})(x_i - x_{i+1})\cdots(x_i - x_n)}$$

لاحظ أن $f_i(x_i) = 1$ و $f_i(x_j) = 0$ حيث x_j قيمة x فى الجدول وتختلف عن x_i . ولذلك فاننا نؤكد أن $y(x_i) = y_i$

اكتب برنامج بيسك ليقرأ أزواجاً من البيانات مقدارها n حيث لا تتعدى n القيمة 10 ، ثم بعد ذلك احصل على قيمة مستكملة للمتغير y عند نقطة محددة أو أكثر من قيم x . استخدم البرنامج لتحصل على قيم مستكملة للمتغير y عند $x = 13.7$ و $x = 37.2$ و $x = 112$ ، $x = 147$ من البيانات الموجودة فى الجدول ٥ – ٦ . وقرر كم عدد أزواج البيانات المجدولة والمطلوبة فى تمرين الحسابات من أجل الحصول على قيمة مستكملة وصحيحة ومعقولة للمتغير y .

جـدول ٥ — ٦

y	0.21073	0.37764	0.45482	0.49011	0.50563	0.49245	0.47220	0.43433	0.33824	0.19390
x	0	10	20	30	40	50	60	80	120	180

(ط) يصف المثال ٤ — ٥ طريقة **التعويض المتعاقب** لحل معادلة جبرية للصيغة $x = F(x)$ بواسطة طريقة فنية للتكرار تستخدم الصيغة التكرارية $x_{i+1} = F(x_i)$.

طريقة أخرى ، وغالباً ما تكون أكثر فعالية ، لحل المعادلات من هذا الطراز وهى تكرار **نيوتن — رابسون** *Newton - Raphson* (وتسمى أحياناً **طريقة نيوتن**) ولاستخدام هذه الطريقة ، يجب أن تكتب المعادلات الجبرية فى الصيغة $f(x) = 0$ وتستخدم الصيغة التكرارية :

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

حيث تمثل $f'(x_i)$ المشتقة الأولى من $f(x)$ محسوبة عند x_i .

تنفذ التكرارات بنفس طريقة التعويضات المتعاقبة أى تحسب قيمة x_{i+1} من الصيغة العكسية ثم تقارن بقيمة x_i . إن لم تكن القيمتان متقاربتين تقارباً كافياً ، تستخدم قيمة x_{i+1} للتعويض فى الجانب الأيمن من المعادلة التكرارية وتكرر الحسابات مرة أخرى .

اكتب برنامج بيسك لحل معادلات جبرية غير خطية بأى من الطرق الفنية السابقة . قرر أى طريقة سوف تستخدم وذلك بتحديد قيمة رقية مناسبة لمتغير إدخال .

استخدم البرنامج لحل المعادلة $x + \cos x = 1 + \sin x$ لبعض قيم x التى تتراوح قيمها ما بين $\pi/2$ و π . حل باستخدام كلتا الطريقتين الفنيتين . أى طريقة تبدو أفضل ؟

(ى) نفرض أننا أعطينا عدداً من النقط المنفصلة (x_1, y_1) و (x_2, y_2) و . . . . . و (x_n, y_n) حيث تقرأ من منحنى $y = f(x)$ ، وتتراوح قيمة x ما بين x_1 و x_n ونرغب فى حساب المساحة التقديرية الواقعة تحت المنحنى وذلك بتقسيم المنحنى إلى عدد من المستطيلات الصغيرة وحساب مساحات هذه المستطيلات . (وتعرف هذه **بقاعدة شبه المنحرف** (*Trapezoidal rule*) استخدم الصيغة الرياضية :

$$A = \tfrac{1}{2}(y_1 + y_2)(x_2 - x_1) + \tfrac{1}{2}(y_2 + y_3)(x_3 - x_2) + \cdots + \tfrac{1}{2}(y_{n-1} + y_n)(x_n - x_{n-1})$$

لاحظ أن متوسط الارتفاع لكل مستطيل يعطى بواسطة $\tfrac{1}{2}(y_i + y_{i+1})$ وأن العرض لكل مستطيل مساوياً لكمية $(x_{i+1} - x_i)$ حيث تأخذ i القيم 1 و 2 و . . . . . و $(n - 1)$.

استخدم البرنامج ليحسب المساحة تحت المنحنى $y = x^3$ بين حدود $x = 1$ و $x = 4$. حل هذه المسألة أولا بعدد 16 من النقط المتساوية التباعد ، ثم بعد ذلك بعدد 61 نقطة وأخيراً بعدد 301 نقطة — لاحظ أن دقة الحل سوف تأخذ فى التحسن كلما زاد عدد النقط . (الإجابة الصحيحة لهذه المسألة هى 63.75) .

(ك) تصف المسألة ٥ — ٥٧ (ى) عاليه ، طريقة تعرف **بقاعدة شبه المنحرف** (*Trapezoidal rule*) لحساب المساحة تحت المنحنى $y(x)$ ، وحيث تستخدم مجموعة من القيم المجدولة (x_1, y_1) و (x_2, y_2) و . . . . و (x_n, y_n) وذلك لوصف المنحنى . إذا كانت القيم المجدولة متساوية التباعد فان المعادلة المعطاة فى ٥ — ٥٧ (ى) يمكن تبسيطها لقراءة :

$$A = \frac{1}{2}(y_1 + 2y_2 + 2y_3 + 2y_4 + \cdots + 2y_{n-1} + y_n)\,\Delta x$$

حيث Δx هى المسافة ما بين قيمتين متعاقبتين للمتغير x .

طريقة فنية أخرى يمكن تطبيقها عندما يوجد عدد زوجى من المسافات متساوية التباعد ، أى عدداً فردياً لنقط البيانات ، وتعرف بقاعدة سمبسون (*Simpson rule*) . ومعادلة الحسابات لتطبيق قاعدة سيمبسون هى :

$$A = \frac{1}{3}(y_1 + 4y_2 + 2y_3 + 4y_4 + 2y_5 + \cdots + 4y_{n-1} + y_n)\,\Delta x$$

وبالنسبة لأى قيمة معينة من Δx . سوف تنتج قاعدة سيمبسون نتائج أكثر دقة عن تلك التى تنتج من قاعدة شبه المنحرف .

اكتب برنامج بيسك لحساب المساحة تحت منحنى باستخدام كل من الطرق السابقة ، مع فرض رقم فردى من نقط البيانات متساوية التباعد . ثم قرر أياً من هذه الطرق سوف تستخدم وذلك باعطاء قيمة رقية مناسبة لبعض متغيرات الإدخال . اسمح بعدد يصل إلى 101 مجموعة من البيانات ، حيث يمكن لقيم البيانات المجدولة أن تقرأ لحاسب أو أن تحسب داخلياً باستخدام معادلة جبرية .

استخدم البرنامج لحساب المساحة تحت المنحنى :

$$y = e^{-x^2}$$

حيث تتراوح قيم x من صفر إلى واحد . احسب المساحة باستخدام كل من طريقتى الحسابات ، وقارن بين النتائج بالإجابة الصحيحة وهى $A = 0.7468241$.

(ل) يمكن كتابة المعادلة التفاضلية من الدرجة الأولى كما يلى :

$$\frac{dy}{dx} = f(x, y)\,, \quad x \geq x_0$$

مع $y(x_0) = y_0$ ؛ حيث y_0 تمثل قيمة رقية معروفة . والهدف من حل المعادلة التفاضلية هو الحصول على معادلة ، أو مجموعة من القيم المجدولة للمتغير y كدالة فى المتغير x .

يمكن حل معادلة تفاضلية من هذا النوع بواسطة « خطوات متقدمة » متتالية بمسافات صغيرة فى اتجاه x . وبمعنى آخر ، ابدأ بالقيمة المعروفة $y(x_0)$ ، يمكن حساب قيمة $y(x_1)$ حيث $x_1 = x_0 + \Delta x$ و Δx رقم صغير معرف سابقاً (حجم الخطوة) . ثم بعد ذلك يمكن الحصول على قيمة $y(x_2)$ ، حيث $x_2 = x_1 + \Delta x$ ؛ ثم $y(x_3)$ حيث $x_3 = x_2 + \Delta x$ وهكذا حتى يتم حساب عدد كاف من النقط .

أسهل طريق لحساب قيمة $y(x_i + 1)$ عند إعطاء قيمة $y(x_i)$ هو استخدام **طريقة أويلر** (*Euler's method*)

$$y_{i+1} = y_i + f(x_i, y_i)\,\Delta x$$

حيث y_i تمثل $y(x_i)$ و y_{i+1} تمثل $y(x_i + 1)$ و $f(x_i, y_i)$ تمثل dy/dx محسوبة عند (x_i, y_i) . طريقة أويلر طريقة سهلة للعمل بها ، ولكن نتائجها إلى حد ما هى تقريب غير صحيح للمنحى $y(x)$ إلا إذا اختيرت Δx على أنها قيم صغيرة جداً . على ذلك يتطلب عدداً كبيراً من النقط من أجل حساب $y(x)$ على مدى واسع نسبياً للمتغير ، $x_0 \leq x \leq x_n$

هناك طريقة أكثر دقة لحساب قيمة y_{i+1} تعرف **بطريقة رونج كوتا** (*Runge Kutta method*) من الدرجة الرابعة :

حيث

$$y_{i+1} = y_i + \tfrac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$
$$k_1 = f(x_i, y_i)\,\Delta x$$
$$k_2 = f(x_i + \Delta x/2,\ y_i + k_1/2)\,\Delta x$$
$$k_3 = f(x_i + \Delta x/2,\ y_i + k_2/2)\,\Delta x$$
$$k_4 = f(x_i + \Delta x,\ y_i + k_3)\,\Delta x$$

وعلى ذلك بالنسبة لنقطة محددة (x_i, y_i) تتلخص الطريقة فى حساب $x_{i+1} = x_i + \Delta x$ ومنها يتم حساب قيم للمتغيرات k_1 و k_2 و k_3 و k_4 وأخيراً تحدد قيمة y_{i+1} .

اكتب برنامج بيسك يحل معادلة تفاضلية من الدرجة الأولى بشرط إبتدائى محدد باستخدام إحدى الطريقتين السابقتين. ثم قرر أياً من الطريقتين سوف تستخدم باعطاء قيمة رقمية مناسبة لأحد المتغيرات . صف حجم الخطوة (Δx) والعدد الإجمالى للخطوات (n) كدخلات .

استخدم البرنامج فى حل المعادلة التفاضلية :

$$\frac{dy}{dx} = x - y, \quad 0 \leq x \leq 2$$

حيث $y(0) = 1$ ، أوجد حلا باستخدام إحدى الطريقتين السابقتين . قارن النتائج التى تم الحصول عليها بالناتج الصحيح والمعطاة بالمعادلة $y = 2e^{-x} + x - 1$. قرر إلى أى حد يجب أن يكون حجم الخطوة صغيراً لكل طريقة من أجل الحصول على حل صحيح إلى ثلاثة أرقام معنوية .

الجزء الثانى : البيسك المتقدم

الفصل ٦

الدوال والبرامج الفرعية الصغيرة
Functions and Subroutines

٦ ــ ١ تعـريف الدالة ــ جملة DEF
DEFINING A FUNCTION—THE DEF STATEMENT

لتجنب تكرار برمجة نفس الحسابات . يمكن أن يكتب المبرمج الدوال الخاصة به ، لاستخدامها بجانب الدوال المكتبية . تعرف الدالة ذات السطر الواحد بواسطة جملة *DEF* (*DEFINE*) . وتتكون الجملة من رقم السطر ، ثم الكلمة الدالة DEF ثم تعريف الدالة . ويتكون تعريف الدالة نفسها من إسم للدالة يتبعه علامة التساوى ثم يتبعه ثابت مناسب أو متغير مناسب أوصيغة رياضية . وإذا تطلبت الدالة خلاصات فيجب أن تظهر فوراً بعد إسم الدالة ، محصورة بين قوسين ومنفصلة عن بعضها بواسطة فصلات (,) . ويسمح باستخدام المتغيرات التى ليس لها أدلة فقط كخلاصات فى تعريف الدالة .

يمكن أن تعرف كل من الدوال الرقية والدوال الحرفية بجملة DEF (تعطى الدالة الرقية قيمة عددية ، أما الدالة الحرفية فتعطى قيمة سلسلة حرفية) . إذا كانت الدالة رقية فيجب أن يتكون إسمها من ثلاثة حروف ، حيث أول حرفين يجب أن يكونا FN . وبذلك فيمكن وجود عدد من الدوال يصل إلى 26 دالة منفصلة فى البرنامج الواحد (FNA و FNB وFNC ....) .

ويجب أن يتكون إسم الدالة الحرفية من ثلاثة حروف ويتبعها علامة الدولار ($) ومرة أخرى يجب أن يكون أول حرفين هما FN . كذا يمكن تعريف عدد من الدوال الحرفية يصل عددها إلى 26 دالة فى البرنامج الواحد (FNA$ و FNB$ وFNC$ و......) الدالة الحرفية والرقية التى لها نفس الحروف الثلاثة (مثال FNP و FNP$) كل منها كيان منفصل ويمكن أن تظهر فى نفس البرنامج .

مثال ٦ – ١

مبين فيما يلى ثلاثة تعريفات نموذجية لدوال من سطر واحد :

```
10 DEF FNA(X)=X↑3+2*X↑2−3*X+4
20 DEF FNC$="NAME AND ADDRESS:"
30 DEF FNR(A,B,C)=SQR(A↑2+B↑2+C↑2)
```

تعرف الجملة الأولى والثالثة دوال رقية FNA و FNR ، وتعرف الجملة الثانية دالة حروف FNC$. لاحظ أن الدالة الثانيةلا تحتوى على خلاصة . ولاحظ أيضاً أن الدالة الثالثة تستخدم الدالة المكتبية SQR فى تعريف الدالة .

تتطلب بعض نسخ البيسك أن تسبق تعريف الدالة ما يناظرها من مراجع دالة ، وتسمح نسخ أخرى بظهور تعريف الدالة فى أى مكان فى برنامج البيسك وفى كلتا الحالتين . فالأسلوب الجيد لممارسة البرمجة هو تجميع كل تعاريف الدوال ووضعها بقرب بداية البرنامج ، ويسهم ذلك فى تكوين برنامج مرتب واضح ومقروء .

ويجب أن نوضح أن وجود جملة DEF تسهم فقط فى تعريف الدالة . ولحساب قيمة الدالة من المهم أن نشير إلى إسم الدالة فى مكان آخر من البرنامج كما نعمل تماماً مع الدالة المكتبية . وسوف نرى كيف ننجز ذلك فى القسم التالى .

٦ ــ ٢ الاشارة الى الدالة REFERENCING A FUNCTION

يمكن الإشارة إلى الدالة (أى حساب قيمتها) وذلك بذكر إسم الدالة من خلال جملة بيسك ، كما لو كان إسم الدالة متغيراً عادياً . ويجب أن يتبع إسم الدالة مجموعة من الخلاصات محصورة بين أقواس ومفصولة بواسطة فصلات (,) .

عند حساب قيمة الدالة فان قيم الخلاصات تعطى بواسطة إشارة الدالة وليس تعريف الدالة . ولهذا السبب ، فان الخلاصات التى تظهر فى جملة DEF تسمى خلاصات زائفة . وأسماء الخلاصات فى الإشارة للدالة لا تستلزم أن تكون هى نفسها فى تعريف الدالة . بينما يجب أن يكون للخلاصات نفس العدد ، ويجب أن تكون الخلاصات من النوع الصحيح (أى عددى أو حرفى) .

مثال ٦ - ٢

يبين فيما يلى تكوين هيكل لبرنامج بيسك يحتوى إشارتين لدالة معرفة بواسطة المبرمج .

```
10 DEF FNA(X)=X↑3+2*X↑2-3*X+4
   ...
50 LET W=FNA(Y)+Z
   ...
90 IF FNA(C)>=C1 THEN 140
```

يترتب على الجملة رقم 50 حساب قيمة الدالة FNA باستخدام القيمة الحالية للمتغير Y كعامل مدخل . (إذن فان الدالة سوف تعطى القيمة Y↑3+2*Y↑2-3*Y+4) وفى الجملة رقم 90 تحسب قيمة نفس الدالة ولكن باستخدام القيمة الحالية C (وبذلك تعطى القيمة C↑3+2*C↑2-3*C+4) .

لاحظ أن تعريف الدالة كان يمكن أن يلى الإشارة للدالة بدلا من أن يسبقها . يجب أن تناظر الخلاصات فى مرجع الدالة للخلاصات الزائفة على أساس واحد لواحد (أى تناظر كل خلاصة الخلاصة الزائفة) عندما يراد أكثر من خلاصة واحدة . ومرة أخرى يجب أن يكون التناظر بالنسبة لعدد الخلاصات ونوع كل خلاصة ، ولكن ليس بالنسبة لأسماء الخلاصات .

مثال ٦ - ٣

يحتوى برنامج بيسك على الجمل التالية :

```
 80 LET A=FNR(C,X,Y)
    ...
250 DEF FNR(A,B,C)=SQR(A↑2+B↑2+C↑2)
```

سوف تسبب جملة LET (السطر رقم 80) فى حساب قيمة الدالة SQR(C↑2+X↑2+Y↑2) وبعد ذلك تعطى النتيجة للمتغير A .

لاحظ أن الإشارة للدالة (السطر رقم 80) وتعريف الدالة (السطر رقم 250) يحتوى كل على 3 خلاصات رقية ولكن أسماء الخلاصات لا تتناظر .

وبالرجوع إلى القسم ٦ - ١ تذكر أن أسماء الخلاصات التى تظهر فى تعريف الدالة (أى الخلاصات الزائفة) يجب أن تكون متغيرات بدون أدلة . ولكن لنا مطلق الحرية فى الإشارة للدالة . فهنا يمكن كتابة الخلاصات كثوابت وكتغيرات ذات أدلة أو صيغاً رياضية أو إشارات لدوال أخرى . فى الحقيقة قيمة كل خلاصة هى التى تستخدم فى الحسابات .

مثال ٦ - ٤

يحتوى برنامج بيسك على الجمل التالية :

```
30 DEF FNR(A,B,C)=SQR(A↑2+B↑2+C↑2)
   ...
170 LET X3=FNR(K(I),5*(P+Q),LOG(T))
```

لاحظ أن تعريف الدالة يحتوى فقط على متغيرات بدون أدلة كخلاصات . ولكن ، فى الإشارة إلى الدالة ، نرى أنه يمكن التعبير عن الخلاصات كتغيرات ذات أدلة أو صيغاً رياضية أو إشارات لدوال مكتبية . وتنفيذ البرنامج يتسبب فى حساب قيمة الدالة .

```
SQR(K(I)↑2+(5*(P+Q))↑2+LOG(T)↑2)
```

وتعطى قيمة الناتج بعد ذلك للمتغير X3 .

لا يستلزم أن تتقيد المتغيرات المستخدمة فى تعريف الدالة بالخلاصات . فيمكن لأى متغيرات أخرى للبرنامج أن تظهر أيضاً (متضمنة متغيرات ذات أدلة) . وعند حساب قيمة الدالة فانها تستخدم القيم الأكثر حداثة التى أعطيت لهذه المتغيرات .

مثال ٦ — ٥

فيما يلى تكوين هيكل لبرنامج بيسك .

```
30 DEF FNZ(X,Y)=(C1*X+C2*Y)/(C1+C2)
   ...
60 LET C1=10
70 LET C2=20
80 LET R=FNZ(P,Q)
```

تنفيذ الجملة رقم 80 سوف يتسبب فى حساب قيمة الدالة :

```
(10*P+20*Q)/(10+20)
```

لاحظ أن القيم C1 و C2 لا تعطى كخلاصات . وتستخدم القيم الأكثر حداثة التى تم إعطاؤها للمتغيرين C1 و C2 (أى $C1 = 10$ ، $C2 = 20$) عند حساب قيمة الدالة .

فى مثال ٦ — ٦ التالى نرى توضيحاً مفصلا لاستخدام دالة معرفة بواسطة المبرمج .

مثال ٦ — ٦ البحث عن أقصى قيمة Search for a Maximum

نفرض أننا نرغب فى إيجاد قيمة x التى يترتب عليها أن تزداد الدالة :

$$y = x \cos x$$

إلى الحد الأعلى وذلك خلال المدى المحصور بين $x = 0$ ناحية اليسار و $x = \pi$ ناحية اليمين . وسوف تتطلب معرفة قيمة x بكل دقة . وسوف نطلب أيضاً أن يكون مخطط البحث ذا كفاءة عالية نسبياً بمعنى أنه يجب أن تحسب الدالة $y = x \cos x$ أقل عدد ممكن من المرات .

ولحل هذه المسألة يمكن أن تولد أرقام كثيرة لدوال تجريبية متقاربة المسافات (وهى حساب قيمة الدالة عند $x=0$ وعند $x=0.0001$ وعند $x = 0.0002$ و . . . و $x = 3.1415$ و $x = 3.1416$) ثم نحدد أكبر القيم وذلك بالفحص المرئى . ولن تكون كفاءة هذه الطريقة عالية ، لأنها تتطلب تدخلا إنسانياً للحصول على النتيجة النهائية . وبدلا من ذلك دعنا نستخدم **مخطط الحذف** وهو إجراء حسابى له فاعليه عالية لكل الدوال والتى لها « نقطة قصوى » واحدة فقط خلال مدى البحث .

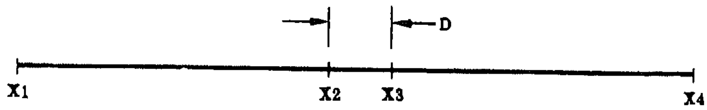
طريقة اجراء الحسابات Computational Procedure

نفرض أننا وضعنا نقطتى بحث عند منتصف المدى ، متباعدة بمقدار مسافة صغيرة جداً عن بعضها كما هو مبين فى شكل ٦ – ١ ، حيث :

X1 = النهاية اليسرى لمدى البحث .
X2 = نقطة البحث الداخلية ناحية اليسار .
X3 = نقطة البحث الداخلية ناحية اليمين .
X3 = النهاية اليمنى لمدى البحث .
D = المسافة ما بين X2 و X3 .

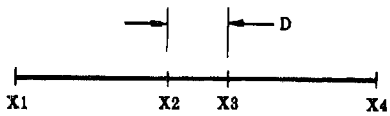
إذا تم تعريف X1 و X2 و D فان النقط الداخلية يمكن حسابها :

```
X2=X1+.5*(X4-X1-D)
X3=X1+.5*(X4-X1+D)=X2+D
```



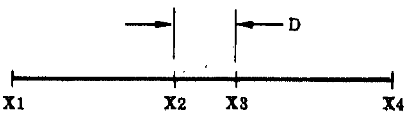
شكل ٦ – ١

دعنا نحسب قيمة الدالة $y = x\cos x$ عند X2 وعند X3 ودعنا نسمى هذه القيم Y2 و Y3 على الترتيب . نفرض أن Y2 قيمتها أكبر من Y3 وبذلك نعلم أن أقصى قيمة نبحث عنها تقع فى أى مكان بين X1 و X3 وبذلك نحتفظ بهذا الجزء فقط لمدى البحث والذى يتراوح ما بين $x = X1$ إلى $x = X3$ (والآن سوف نشير للنقطة القديمة X3 كما لو كانت X4 ، حيث أنها النهاية اليمنى لمدى البحث الجديد) ثم نولد نقطتى بحث جديدتين X2 و X3 . وسوف توضع هذه النقط فى منتصف مدى البحث القديم ، بينهما المسافة D ، كما هو مبين فى شكل ٦ – ٢ .



سابقاً X3

شكل ٦ – ٢



سابقاً X2

شكل ٦ – ٣

ومن ناحية أخرى ، نفرض أن قيمة Y3 فى مدى البحث الأصل كانت أكبر من قيمة Y2 فان ذلك يشير إلى أى مدى يجب أن يقع البحث الجديد ما بين X2 و X4 وبذلك فاننا نعيد تسمية النقطة المسماة أصلا X2 بالإسم الجديد X1 ونولد نقطتى بحث جديدتين X2 و X3 عند منتصف مدى البحث الجديد كما هو مبين فى شكل ٦ – ٣ .

نستمر فى توليد زوج جديد من نقط البحث عند منتصف كل مدى جديد ، ثم نقارن بين قيم y ، ثم نحذف جزءاً من مدى البحث حتى يصبح المدى أقل من D * 3 وعند حدوث ذلك لا يمكننا التفرقة بين النقط الداخلية وبين حدود المدى . من م فاننا قد وصلنا إلى نهاية البحث .

عندما تقارن قيم Y2 و Y3 فى كل مرة نحذف من مدى البحث الجزء الذى يحتوى على القيمة الأصغر للمتغير y . أما إذا كانت القيمتان الداخليتان للمتغير y متساويتان (ويمكن أن يحدث ذلك ، بالرغم أنه غير عادى) ، عندئذ سوف يتوقف إجراء البحث ونفترض أن النقطة القصوى تحدث عند منتصف نقطتى البحث .

فور انتهاء البحث ، سواء كان ذلك لأن مدى البحث أصبح صغيراً صفراً كافياً أو لأن النقطتين الداخليتين تعطيان قيما متطابقة لـ y فيمكننا حساب أقصى موقع بالتقريب على أنه :

X5 = 0.5*(X2 + X3)

ويمكن الحصول على القيمة القصوى المناظرة للدالة من X5*COS (X5)

المخطط التمهيدى للبرنامج The Program Outline

١ – عرف الدالة $y = x \cos x$.

٢ – اقرأ القيم البدائية للمتغيرين X1 و X2 والمسافة D .

٣ – اجعل قيمة I = 1 (حيث I عداد داخلى) .

٤ – احسب زوجاً من النقط الداخلية .

٥ – اكتب قيم x عند نهاية المدى وعند النقط الداخلية واكتب قيم y المناظرة لها .

٦ – قارن بين X2 و X3 .

(أ) إذا كانت Y2 أكبر من Y3 ، فبدل أسماء X3 و X4 ، وبذلك تعرف مدى جديداً للبحث وتقدم للخطوة رقم 7 .

(ب) إذا كانت Y3 أكبر من Y2 ، فبدل أسماء X1 و X2 ، وبذلك تعرف مدى جديداً للبحث وتقدم للخطوة رقم 9 .

(ج) إذا كانت Y2 تساوى Y3 تقدم للخطوة رقم 9 .

٧ – اختبر لترى ما إذا كانت I = 100 وإذا تحقق ذلك اكتب رسالة ملائمة وتوقف . وإلا زد عداد التكرار واستمر .

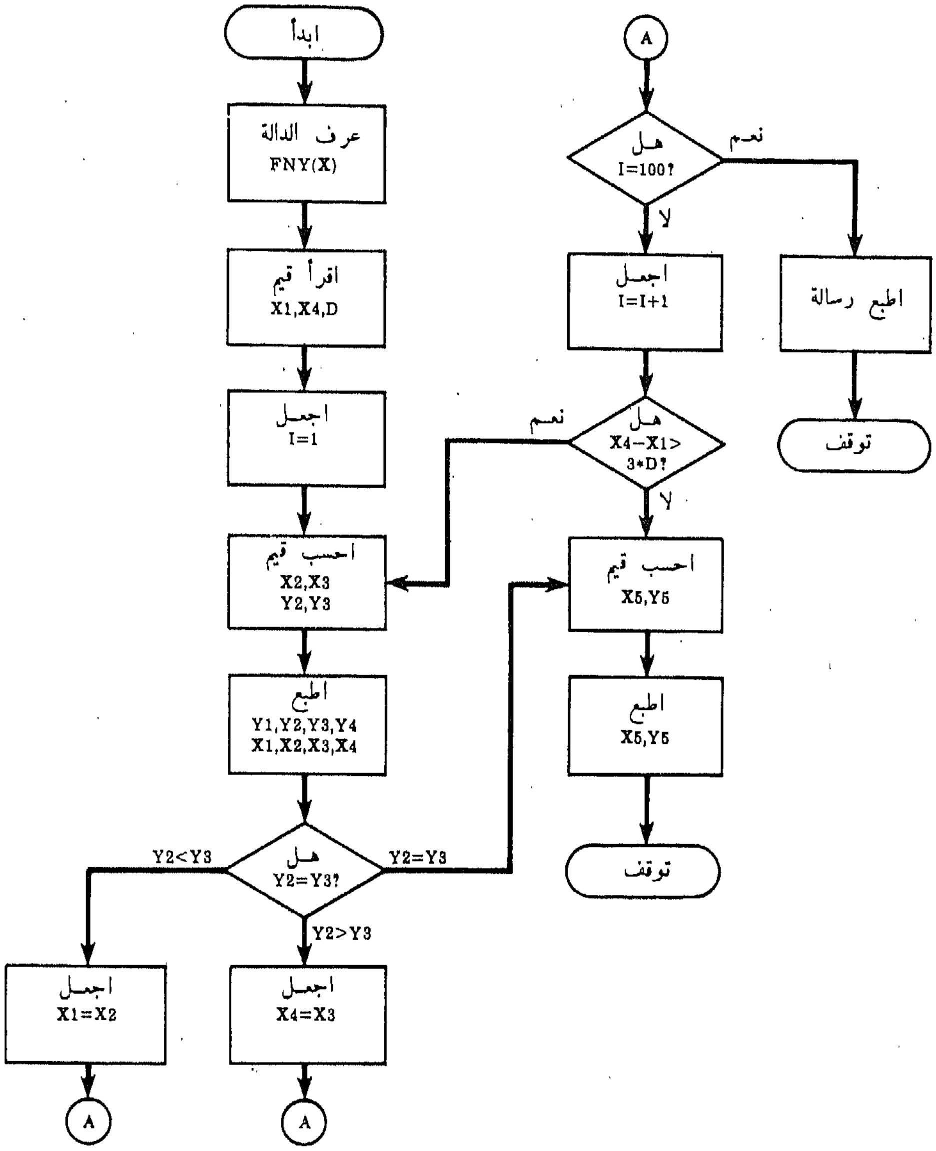
٨ – اختبر لترى ما إذا كانت (X4 — X1) < 3*D وإذا تحقق ذلك ارجع مرة ثانية للخطوة رقم 4 وإلا أكمل للخطوة التالية .

٩ – احسب قيمة X5 = 0.5*(X2 + X3)

Y5 = X5*COS (X5)

وبعد ذلك أكتب النتائج النهائية وتوقف .

مبين فى شكل ٦ – ٤ خريطة سير عمليات لهذا الإجراء .



شكل ٦ – ٤

برنامج البيسك The BASIC Program

يظهر برنامج البيسك الحقيقي في شكل ٦ – ٥ . وللتبسيط فاننا ضمنا دالة معرفة لحساب قيمة الكمية $y = x \cos x$ (لاحظ أن تعريف الدالة يتضمن الإشارة إلى الدالة المكتبية COS) . ولاحظ أننا أشرنا إلى الدالة في السطور 170 و 180 و 200 و 440 . وتعطى قيما مختلفة للخلاصة في كل إشارة للدالة (أي قيم مختلفة للمتغير x) .

ويبين شكل ٦ – ٦ المخرجات المولدة بواسطة البرنامج عندما $X1 = 0$ و $X4 = 3.14159$. $D = 0.0001$. ورأينا أن أقصى قيمة للمتغير y هي 5.5611 تقريباً وتحدث عند $x = 0.8604$. لاحظ أنه تم الحصول على هذه النتيجة لدرجة عالية من الدقة بعدد من نقط البحث وصلت إلى 14 قيمة !

ويمكن أيضاً استخدام الإجراء المعطى فى هذا المثال للحصول على أقل قيمة للدالة x. فى الحقيقة، يمكن استخدام البرنامج نفسه بعد إجراء تعديلات طفيفة. تساعدنا هذه الطريقة للحصول على أقل قيمة بطريقة فنية عالية الكفاءة لحساب جذور المعادلة الجبرية غير الخطية. فمثلا، نفرض أننا نريد إيجاد قيمة x التى تجعل $f(x)$ مساوية للصفر. أحد الأمثلة لذلك: $f(x) = x - \sin x + \cos x - 1$ إذا جعلنا $y(x) = f(x)^2$ فان الدالة $y(x)$ سوف تكون دائماً موجبة ما عدا قيم x التى هى جذور الدالة المعطاة، أى عندما تكون $y(x) = 0$ وبذلك أى قيمة تجعل $y(x)$ لها أقل قيمة سوف تكون أيضاً جذراً للمعادلة $f(x) = 0$.

```
10 REM SEARCH FOR A MAXIMUM OF THE FUNCTION Y=X*COS(X)
20 DEF FNY(X)=X*COS(X)
30 PRINT "LEFT END OF INTERVAL (X1) =";
40 INPUT X1
50 PRINT
60 PRINT "RIGHT END OF INTERVAL (X4) =";
70 INPUT X4
80 PRINT
90 PRINT "MINIMUM SEPARATION BETWEEN INTERIOR POINTS (D) =";
100 INPUT D
110 LET I=1
120
130 REM CALCULATE INTERIOR POINTS
140
150 LET X2=X1+.5*(X4-X1-D)
160 LET X3=X2+D
170 LET Y2=FNY(X2)
180 LET Y3=FNY(X3)
190 PRINT
200 PRINT "Y1=";FNY(X1),"Y2=";Y2,"Y3=";Y3,"Y4=";FNY(X4)
210 PRINT "X1=";X1,"X2=";X2,"X3=";X3,"X4=";X4
220 IF Y2<Y3 THEN 300
230 IF Y2=Y3 THEN 400
240
250 REM Y2 GREATER THAN Y3 - RETAIN LEFT INTERVAL
260
270 LET X4=X3
280 GOTO 340
290
300 REM Y3 GREATER THAN Y2 - RETAIN RIGHT INTERVAL
310
320 LET X1=X2
330
340 REM TEST FOR END OF SEARCH
350
360 IF I=100 THEN 470
370 LET I=I+1
380 IF (X4-X1)>3*D THEN 130
390
400 REM COMPUTE FINAL SOLUTION
410
420 LET X5=.5*(X2+X3)
430 PRINT
440 PRINT ,"XMAX=";X5,"YMAX=";FNY(X5)
450 STOP
460
470 REM TERMINATE COMPUTATION BECAUSE OF MAXIMUM ITERATION COUNT
480
490 PRINT "MAXIMUM NUMBER OF ITERATIONS EXCEEDED - COMPUTATION ENDS"
500 END
```

شكل ٦ — ٥

٦ — ٣ الدوال المتعددة السطور MULTILINE FUNCTIONS

يوجد عديد من الحسابات التى لا يمكن إجراؤها أو القيام بها فى سطر واحد. ويمكن أن يكون ذلك صحيحاً وخصوصاً فى الحسابات التى تحتوى على صيغ رياضية طويلة أو على عمليات تفرع مشروط. وتدعم بعض نسخ البيسك صيغه الدالة المتعددة السطور والتى تناسب الحاسبات من هذا النوع.

يمكن للدالة المتعددة السطور (مثل الدالة ذات السطر الواحد) ، أن تحتوى على أى عدد من الخلاصات الزائفة كمدخلات ولكنها ترجع قيمة واحدة فقط . يجب أن تكون أول جملة هى جملة DEF . ومع ذلك على عكس الدالة ذات السطر الواحد ، فلا يتضمن تعريف الدالة فى جملة DEF . ويجب أن تكون جملة *FNEND* أى (*FUNCTION END*) آخر جملة وتتكون ببساطة من رقم السطر وتتبعه الكلمة الدالة FNEND .

ويمكن أن يوجد بين جملتى DEF و FNEND أى عدد من الجمل والتى تعرف الدالة . ويجب أن تعطى إحدى هذه الجمل قيمة لإسم الدالة . ويتم إنجاز ذلك عادة بجملة LET حيث يظهر إسم الدالة على يسار علامة التساوى (=) . وتستخدم نفس طريقة التسمية التقليدية كما فى الدالة ذات السطر الواحد .

```
LEFT END OF INTERVAL (X1) =?0

RIGHT END OF INTERVAL (X4) = ?3.14159

MINIMUM SEPARATION BETWEEN INTERIOR POINTS (D) = ?.0001

Y1= 0            Y2= 8.06277E-5                    Y3=-7.64780E-5
Y4=-3.14159
X1= 0            X2= 1.57075      X3= 1.57085      X4= 3.14159

Y1= 0            Y2= 0.555356     Y3= 0.555372     Y4=-7.64780E-5
X1= 0            X2= 0.785372     X3= 0.785473     X4= 1.57085

Y1= 0.555356     Y2= 0.450865     Y3= 0.450795     Y4=-7.64780E-5
X1= 0.785372     X2= 1.17806      X3= 1.17816      X4= 1.57085

Y1= 0.555356     Y2= 0.545438     Y3= 0.545412     Y4= 0.450795
X1= 0.785372     X2= 0.981716     X3= 0.981816     X4= 1.17816

Y1= 0.555356     Y2= 0.560534     Y3= 0.560529     Y4= 0.545412
X1= 0.785372     X2= 0.883544     X3= 0.883644     X4= 0.981816

Y1= 0.555356     Y2= 0.560405     Y3= 0.56041      Y4= 0.560529
X1= 0.785372     X2= 0.834458     X3= 0.834558     X4= 0.883644

Y1= 0.560405     Y2= 0.561094     Y3= 0.561095     Y4= 0.560529
X1= 0.834458     X2= 0.859001     X3= 0.859101     X4= 0.883644

Y1= 0.561094     Y2= 0.560972     Y3= 0.560969     Y4= 0.560529
X1= 0.859001     X2= 0.871273     X3= 0.871373     X4= 0.883644

Y1= 0.561094     Y2= 0.561072     Y3= 0.561071     Y4= 0.560969
X1= 0.859001     X2= 0.865137     X3= 0.865237     X4= 0.871373

Y1= 0.561094     Y2= 0.561093     Y3= 0.561093     Y4= 0.561071
X1= 0.859001     X2= 0.862069     X3= 0.862169     X4= 0.865237

Y1= 0.561094     Y2= 0.561096     Y3= 0.561096     Y4= 0.561093
X1= 0.859001     X2= 0.860535     X3= 0.860635     X4= 0.862169

Y1= 0.561094     Y2= 0.561096     Y3= 0.561096     Y4= 0.561096
X1= 0.859001     X2= 0.859768     X3= 0.859868     X4= 0.860635

Y1= 0.561096     Y2= 0.561096     Y3= 0.561096     Y4= 0.561096
X1= 0.859768     X2= 0.860152     X3= 0.860252     X4= 0.860635

Y1= 0.561096     Y2= 0.561096     Y3= 0.561096     Y4= 0.561096
X1= 0.860152     X2= 0.860343     X3= 0.860443     X4= 0.860635

                 XMAX= 0.860393                    YMAX= 0.561096
```

شكل ٦ — ٦

مثال ٦ — ٧

موضح فيما يلى التخطيط الهيكلى للدالة المتعددة السطور :

```
200 DEF FNA(X,Y,Z)
     ...
250 LET FNA=...
260 FNEND
```

تسمى هذه الدالة FNA وتستخدم خلاصات زائفة X و Y و Z كدخلات . وتعطى قيمة الدالة عند حسابها فى الجملة رقم 250 .

القواعد اللغوية التى تطبق على الدوال المتعددة السطور هى نفسها التى تطبق على الدوال ذات السطر الواحد ؛ (فمثلا يمكن أن يظهر تعريف الدالة فى أى مكان من البرنامج ، ونشير إلى الدالة بتحديد إسمها تتبعه قائمة من الخلاصات محصورة بين قوسين وتفصل بينها بواسطة فصلة (,) و . . . الخ) وبالإضافة إلى ذلك ، فلا يمكن تحويل التحكم بين جملة بداخل الدالة ونقطة خارج نطاق الدالة .

مثال ٦ — ٨

مبين فيما يلى جزء من برنامج بيسك يحتوى على تعريف دالة متعددة السطور إلى هذه الدالة . الغرض من هذه الدالة هو تحديد أقل قيمة لأزواج من الأعداد .

```
 20 DEF FNM(A,B)
 30 LET FNM=A
 40 IF A<=B THEN 60
 50 LET FNM=B
 60 FNEND
    ...
150 PRINT FNM(FNM(C1,C2),FNM(C2,C3))
```

لاحظ تداخل الدالة FNM مع نفسها فى رقم الجملة 150 . وتسبب هذه الجملة إيجاد أصغر قيمة من ثلاث كيات ممثلة بالمتغيرات C1 و C2 و C3 ثم طباعتها .

يمكن ظهور متغيرات أخرى غير المعطاة كخلاصات فى الدالة المتعددة السطور تماماً كما فى الدالة ذات السطر الواحد . ويمكن أن تكون هذه المتغيرات إما متغيرات ذات أدلة أو متغيرات بدون أدلة . وتستخدم القيمة الحالية لهذه المتغيرات عند تنفيذ الدالة كل مرة .

مثال ٦ — ٩

برنامج بيسك يحتوى على تعريف الدالة متعددة السطور التالية :

```
100 DEF FNY(X)
110 IF X>800 THEN 140
120 LET FNY=A+B*X+C*X↑2
130 GO TO 150
140 LET FNY=D+E*X+F*X↑2
150 FNEND
```

لاحظ أن قيمة X تستمد من خلال خلاصة وذلك عند الإشارة إلى الدالة ، بينما قيم A و B و C و D و E و F لا تستمد كخلاصات . وبذلك فان أحدث قيم لهذه المتغيرات سوف تستخدم فى كل مرة تحسب فيها الدالة .

يجب أن يكون مفهوماً أن الدالة التى تحتوى على خلاصات من نوع معين يمكن أن تنتج قيمة من نوع مختلف (مثال ، الدالة التى لها خلاصات حرفية يمكن أن تستخدم لتحديد قيمة رقية) . وعلاوة على ذلك ، لا يستلزم أن تكون الخلاصات من نفس النوع (أى يمكن وجود مزيج منخلاصات رقية وحرفية) . وهذا صحيح لكل من الدالة ذات السطر الواحد والدالة المتعددة الأسطر ومع ذلك تذكر ، أن الخلاصات فى الإشارة للدالة يجب أن تناظر الخلاصات فى تعريف الدالة من حيث العدد والنوع .

مثال ٦ — ١٠

يمثل الهيكل التخطيطى التالى دالة متعددة الأسطر تتطلب خلاصتين إحداهما رقية والأخرى حرفية . والدالة نفسها تعطى نتيجة حرفية (لوجود علامة الدولار فى إسمها) .

```
100 DEF FNW$(C,N$)
    ...
140 LET FNW$=...
150 FNEND
```

عند الإشارة إلى هذه الدالة يجب أن نمدها بخلاصتين إحداهما رقية والأخرى حرفية ، ويجب أن تكونا بهذا الترتيب . لا يستلزم أن تكون أسماء الخلاصات بالطبع نفس أسماء الخلاصات الزائفة . من ثم فيمكن أن يكون مرجع الدالة المناسب مثل :

```
250 LET N$=FNW$(X,T$)
```

توضح الأمثلة ٦ – ١٥ ، ٦ – ٢٠ استخدام الدوال المتعددة الأسطر فى برامج بيسك كاملة .

٦ – ٤ تكويد وفك شفرة البيانات – جملة CHANGE

ENCODING AND DECODING DATA—THE CHANGE STATEMENT

عندما تمثل سلسلة حرفية للحاسب ، فان الحروف التى تكون هذه السلسلة لا تخزن فى الحاسب كحروف ولكنها تخزن كأرقام مكودة بتسلسل كل رقم أو حرف أو حرف خاص يمكن تمثيله برقم فريد خاص به .

توجد عدة أنظمة تكويد رقية مختلفة مستخدمة فى الأجهزة المتنوعة . والنظام الأكثر شيوعاً هو نظام ASCII ذو الأرقام السبعة الثنائيه(+) ، حيث يمثل الحرف A بالرقم (العشرى) 65 ويمثل الحرف B بالرقم 66 ، . . . وهكذا . يبين جدول ٦ – ١ كود ASCII المكافئ لأكثر الحروف شيوعاً .

يتم القيام بعملية التحويل من حروف إلى أرقام ، وبالعكس ، أتوماتيكيا بداخل الحاسب حتى أن المبرمج لا يشعر حقيقة بهذا التحويل ولا حتى حدوثه . ولكن ، فى بعض الأحوال يكون المطلوب هو العمل بالرقم المكافئ للحروف فى السلسلة الحرفية ، ويسمح ، ذلك بالتعامل مع كل حرف على حدة . وجملة *CHANGE* تسمح بالقيام بعمل هذا التحويل .

توجد طريقتان مختلفتان يمكن أن تكتب بهما جملة CHANGE . تتكون الأولى من رقم السطر ثم الكلمة الدالة CHANGE ثم متغير حرفى ثم الكلمة الدالة TO ثم قائمة رقية ويجب أن تكون بهذا الترتيب . وتسبب هذه الجملة تحويل كل حرف من السلسلة الحرفية إلى الرقم المكافئ له ويخزن فى القائمة الرقية . سوف يشير أول عنصر من القائمة (ذو الدليل صفر) إلى عدد الحروف المكودة والتى تحتويها القائمة .

مثال ٦ – ١١

يحتوى برنامج بيسك على الجمل التالية :

```
100 CHANGE N$ TO N
```

حيث N إسم القائمة الرقية نفرض أن N$ ممثلة بالسلسلة MONDAY وأن كود ASCII المبين فى جدول ٦ – ١ قابل

(+) كود أمريكى قياسى لتبادل المعلومات ، وهو كود واسع الانتشار يؤيده المعهد القومى الأمريكى للمعايرة .
(American Standard Code for Information Interchange)

جدول ٦ – ١ كود ASCII ذو الأرقام السبعة الثنائية

كود	حرف	كود	حرف
48	0	65	A
49	1	66	B
50	2	67	C
51	3	68	D
52	4	69	E
53	5	70	F
54	6	71	G
55	7	72	H
56	8	73	I
57	9	74	J
43	+	75	K
45	–	76	L
47	/	77	M
42	*	78	N
94	↑	79	O
40	(	80	P
41	)	81	Q
60	<	82	R
62	>	83	S
61	=	84	T
63	?	85	U
36	$	86	V
34	"	87	W
44	,	88	X
46	.	89	Y
59	;	90	Z
13	عودة العربة		
10	تغذية أسطر		
32	حرف خال		

للاستعمال . وسوف يسبب تنفيذ جملة CHANGE إعطاء القيم التالية لعناصر القائمة N .

(يشير إلى أن عدد الحروف في السلسلة = ٦) $N(0) = 6$

(القيمة العددية المكافئة للحرف M) $N(1) = 77$

(القيمة العددية المكافئة للحرف O) $N(2) = 79$

(القيمة العددية المكافئة للحرف N) $N(3) = 78$

(القيمة العددية المكافئة للحرف D) $N(4) = 68$

(القيمة العددية المكافئة للحرف A) $N(5) = 65$

(القيمة العددية المكافئة للحرف Y) $N(6) = 89$

والآن فن الممكن التوصل للقيمة العددية المكافئة لأى حرف من السلسلة بسهولة وذلك بالرجوع إلى الدليل المناسب) .

يمكن تبادل موضع المتغير الحرفى والقائمة الرقية فى جملة CHANGE . أى أنه يمكن كتابة الجملة كرقم جملة تتبعها الكلمة الدالة CHANGE ثم قائمة رقية ثم الكلمة الدالة TO ثم متغير حرفى . والجملة بهذا الشكل تحول عنصر القائمة الرقية إلى سلسلة حرفية . التحويل إلى حروف يبدأ من العنصر الثانى للقائمة (أى من العنصر ذى الدليل الذى قيمته 1) . وسوف يشير العنصر الأول من القائمة (ذو الدليل صفر) كما سبق إلى عدد الحروف الموجودة فى السلسلة .

مثال ٦ - ١٢

برنامج بيسك يحتوى على الجملة

```
225 CHANGE L TO A$
```

حيث L إسم قائمة رقية . نفرض أن عناصر القائمة L لها القيم التالية والتى تمثل حروف ASCII ذات الأرقام السبعة الثنائية :

```
L(0)=11        L(6)=32
L(1)=83        L(7)=67
L(2)=65        L(8)=76
L(3)=78        L(9)=65
L(4)=84        L(10)=85
L(5)=65        L(11)=83
```

عند تنفيذ جملة CHANGE فسوف يتم تحويل كل عنصر من L مبتدئاً من العنصر L(1) إلى الحرف المقابل والسلسلة الناتجة من ذلك سوف تعطى للمتغير A$. ومن ثم فان المتغير A$ سوف يمثل بالقيمة SANTA CLAUS

فى المثال ٦ - ١٥ سوف نرى برنامج بيسك كاملا يستخدم جملة CHANGE .

٦ — ٥ دالتى ASC و CHR$ THE ASC AND CHR$ FUNCTIONS

تزامل إلى حد بعيد جملة CHANGE الدالتين المكتبيتين ASC و CHR$ الأولى وهى ASC ، تحول أى حرف إلى ما يكافئه من الكود ASCII وبذلك فان الدالة تقبل حرفاً واحداً كخلاصة .

مثال ٦ - ١٣

اعتبر الجملة :

```
50 LET C=ASC(P)
```

وسوف يترتب على هذه الجملة إعطاء القيمة 80 للمتغير C ، حيث 80 فى الكود ASCII تكافئ الحرف P .

وبالمثل الجملة :

```
70 IF L(I)=ASC( ) THEN 110
```

سوف يترتب عليها أن يتحول التحكم إلى رقم الجملة 110 إذا كان المتغير ذو الدليل L (I) له القيمة 32 ، حيث الرقم 32 يكافئ كود ASCII للمسافة الخالية .

الغرض من الدالة المكتبية CHR$ هو عكس الدالة ASC أى أن CHR$ تستخدم لتحويل قيمة كود ASCII المناظر لحرف إلى الحرف نفسه فى هذه الحالة فان قيمة الخلاصة يجب أن تكون كية رقية معروفة لكود ASCII ، والقيمة غير الصحيحة سيتم بترها أوتوماتيكيا .

مثال ٦ – ١٤

اعتبر الجملة :

```
75 LET A$=CHR$(X)
```

إذا كانت القيمة X هى 42 ، فان A$ سوف تمثل الحرف * ، حيث 42 هو تمثيل كود ASCII للحرف (*) .

وبالمثل فان الجملة :

```
310 PRINT CHR$(L(I));
```

سوف يترتب عليها طباعة الحرف P إذا كانت L(I) لها القيمة 80 . دالتى ASC و CHR$ وجملة CHANGE ، متضمنة فى المثال التالى .

مثال ٦ – ١٥ مولد بيجلاتين A Piglatin Generator

بيجلاتين صيغة مكودة بالإنجليزية وتستخدم غالباً كلعبة بواسطة الأطفال . كلمة بيجلاتين تتكون من كلمة إنجليزية بتحوير الصوت الأول (وغالباً ما يكون الحرف الأول) إلى نهاية الكلمة ثم بعد ذلك إضافة الحرف "a" وبذلك فان الكلمة cat تصبح "atca" و "BASIC" تصبح "ASICBA" و piglatin تصبح "iglatinpa" و "igpa atinla" إذا تم هجاؤها كلمتين منفصلتين) ... وهكذا .

ونرغب فى كتابة برنامج بيسك يقبل سطراً من اللغة الإنجليزية ثم يطبع هذا السطر ثانياً مستخدماً طريقة بيجلاتين (Piglatin) .

طريقة إجراء الحسابات Computational Procedure

سوف نفترض أن كل رسالة نصية يمكن طباعتها على سطر واحد يحتوى على 72 حرفاً كما هو متبع على الوحدات الطرفية الخاصة بالمشاركة الزمنية ، مع ترك مسافة خالية بين كل كلمتين متتاليتين . وسوف يكون إجراء الحسابات بعد ذلك مباشراً ويتكون من وسيلة لاستخلاص كل كلمة من الرسالة ، ثم إعادة تنظيم الكلمة ثم إضافة الحرف "a" ثم طباعة الكلمة بعد إعادة تنظيمها . من السهل ، من حيث المبدأ استخلاص كل كلمة من السطر ، حيث تتميز كل كلمة بوجود مكان خال بينها وبين الكلمات الأخرى .

الحسابات التفصيلية خداعة إلى حد ما ، إلا أن مناولة الحروف فى السطر بطريقة فردية (أى كل حرف على حدة) فقط ممكنة إذا حولنا الحروف أولا إلى مكافئاتها الرقمية حسب كود ASCII . وقبل طباعة كل من الكلمات التى أعيد تنظيمها يلزم تغيير الأرقام بكود ASCII مرة أخرى إلى حروف . وسوف تستعمل كل من جملة CHANGE ودالة CHR$ لهذا الغرض .

واستخلاص كميات ASCII التى تمثل كلمة واحدة من القائمة الكاملة لأرقام ASCII يجب أن تتم بكل حذر . سوف نستخدم مؤشرين (متغيرات رقية) لهذا الغرض . وسوف يشير المؤشر الأول إلى موضع ما يكافئ الحرف الأول من الكلمة بالكود ASCII والمؤشر الثانى سوف يشير إلى أين يوجد الحرف الأخير من الكلمة بكود ASCII وسوف يعاد تحديد قيم المؤشرين بعد إعادة تنظيم كل كلمة .

وسوف نستخدم دالة متعددة الأسطر من أجل استخلاص كل كلمة ، أى لتحديد مواضع المؤشرات . ولا توجد أى مشكلة فى تحديد موضع المؤشر الأول ، حيث نعلم أن الكلمة الأولى سوف تبدأ من المكان رقم 1 ، وسوف تبدأ كل كلمة تالية وراء نطاق نهاية الكلمة السابقة بموضعين . بينما لتحديد موضع المؤشر الثانى سوف نفحص كل حرف مكود بكود ASCII وراء نطاق المؤشر الأول حتى نجد مسافة خالية . ومن ثم يتحدد موضع المؤشر الثانى بمكان واحد قبل المسافة الحالية .

المخطط التمهيدى للبرنامج The Program Outline

من أجل كتابة مخطط تمهيدى مفصل لإجراء الحسابات دعنا نعرف أولا الرموز التالية :

N$ = السطر المعطى من النص (سلسلة حرفية) .

L = قائمة رقية تحتوى مكافئات كود ASCII لحروف فى سطر النص (لاحظ أن L سوف تتكون من 72 كمية فردية مكودة) .

P1 = مكان أول حرف من كلمة معينة فى قائمة المكافئات بكود ASCII المخزنة فى L (رقم ما بين 1 إلى 72) .

P2 = مكان آخر حرف من كلمة معينة فى قائمة المكافئات بكود ASCII المخزنة فى L (رقم ما بين 1 إلى 72) .

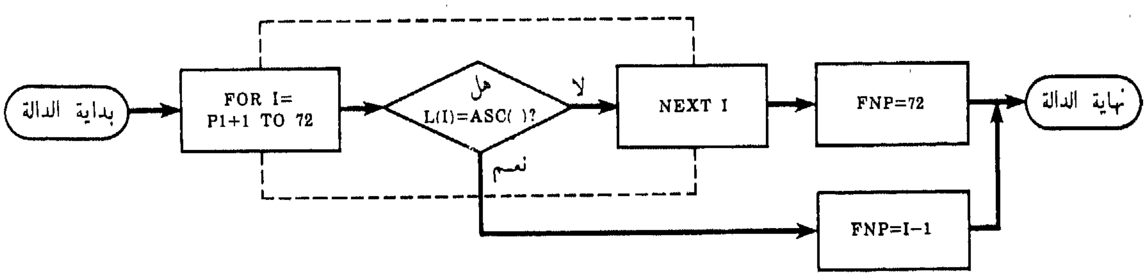
ادرس الدالة متعددة الأسطر FNP والتى ترجع قيمة المتغير P2 عند إعطائها قيمة للمتغير P1 . سوف تستمر الحسابات كما يلى :

١ – لكل قيمة من قيم الدليل I مبتدئاً من I = P1 + 1 ، اختبر لترى إذا كان المتغير ذو الدليل L(I) يحتوى على مكافئ ASCII للمكان الخالى .

(أ) إذا كانت بعض L(I) تمثل مكاناً خالياً (وذلك مشيراً لنهاية الكلمة) ، اجعل P2 = I — 1

(ب) إذا كانت كل L(I) تمثل أى حرف ما عدا المكان الخالى ، اجعل P2 = 72 (نهاية السطر) .

٢ – ارجع قيمة P2 إلى نقطة الإشارة للدالة . شكل ٦ – ٧ يبين خريطة سير عمليات للاجراء الموصوف عالياً .



شكل ٦ – ٧

سوف نواصل بقية البرنامج على النهج التالى :

١ – اقرأ N$

٢ – اختبر لترى ما إذا كانت N$ تساوى الكلمة END . إذا كانت كذلك أنهى الحسابات ، وإلا واصل بالخطوة 3 التالية .

٣ – حدد قيمة كل عنصر من القائمة L بما يكافئ المكان الخالى من الكود ASCII (وبذلك « تمحى » ما يحتمل أن يكون قد تم تخزينه فى L من قبل) .

٤ – حول N$ إلى L

٥ – اجعل P1 = 1

٦ – ارجع للدالة FNP لتكوين قيمة P2 .

٧ – أعـد تنظيم الحروف الموجودة فى الكلمة ثم اطبعها بالطريقة التالية :

(أ) إذا كانت الكلمة تحتوى فقط على حرف واحد ، فان P2 سوف تنطبق مع P1 . من ثم واصل مباشرة من الخطوة (٧ جـ) أدناه .

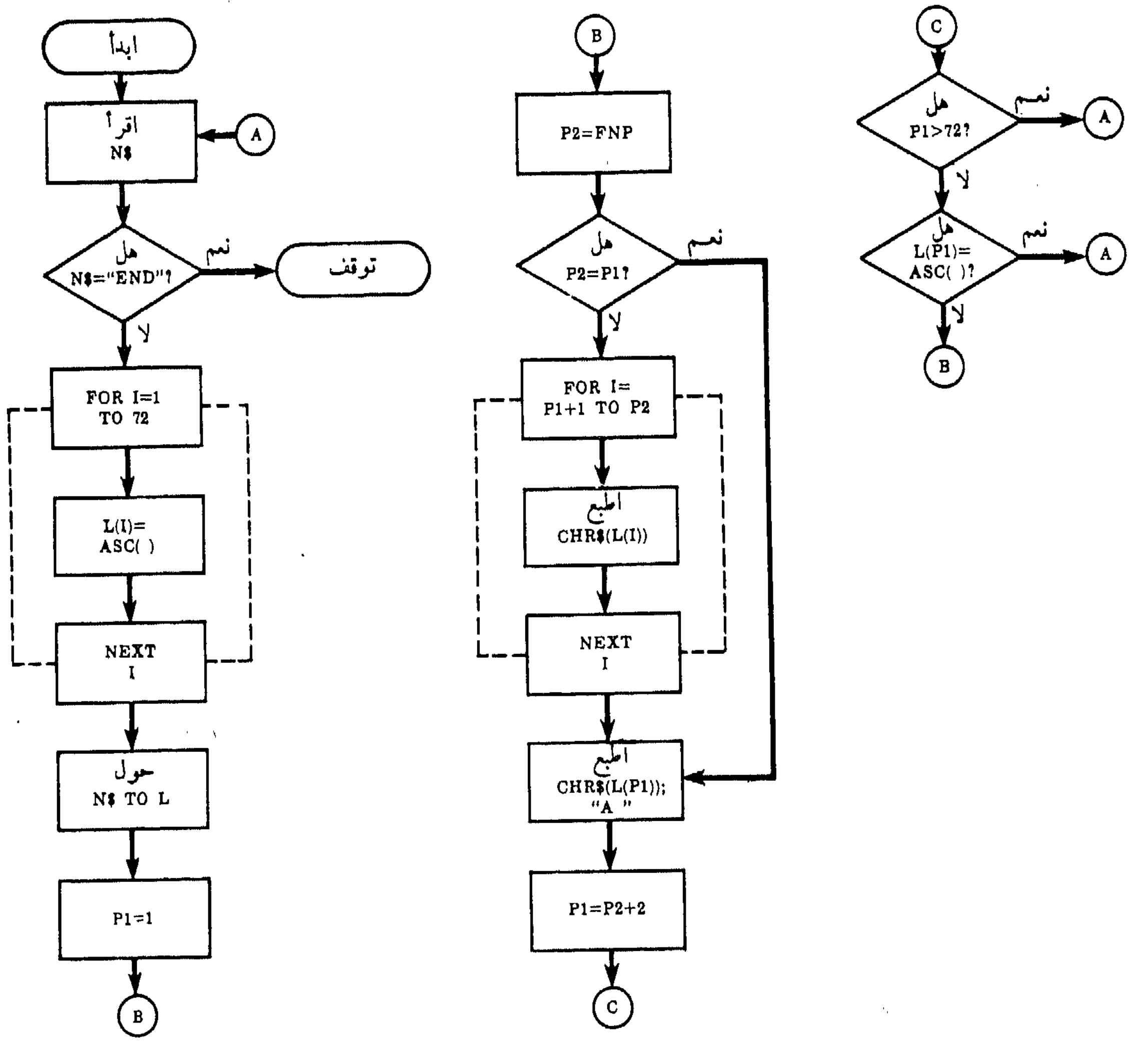
(ب) اطبع الحروف الممثلة بواسطة L (P + 1) إلى L (P2) على التوالى .

(جـ) اطبع الحروف الممثلة بواسطة L (P1) ، ويتبعها فورا الحرف "a" ثم مكان خال .

٨ – كون بداية الكلمة التالية كما يلى :

(أ) اجعـل P1 = P2 + 2

(ب) إذا تعدت قيمة P1 الرقم 72 ، فان سطر النص يكون قد اكتمل ، من ثم ارجع مرة أخرى للخطوة 1 واقرأ سطراً جديداً من النص .



شكل ٦ – ٨

(جـ) إذا كانت L (P1) تمثل سطراً خالياً (حتى إن لم تكن قيمة P1 أكبر من 72) ، وعلى ذلك يكون قد تم الوصول إلى نهاية الرسالة .

(د) اذهب مرة ثانية للخطوة ٦ وكون قيمة جديدة للمتغير P2 .

لاحظ أن هذه الخطة سوف تستمر فى قراءة سطور متتالية من النص حتى يضاف كلمة END (أنظر خطوات ١ ، ٢ عاليه) .

يبين شكل ٦ — ٨ خريطة سير عمليات مناظرة للخطوة سابقة الذكر .

برنامج بيسك The BASIC Program

يحتوى شكل ٦ — ٩ على برنامج البيسك الحقيقى لهذه المسألة . لاحظ استخدام الدالة المكتبية ASC فى السطور 70 و 210 و 390 والدالة CHR$ فى السطور 310 و 330 تستخدم هذه الدوال فى تحويل كل حرف من وإلى المكافآت لها بكود ASCII .

تم برمجة حساب قيمة P2 من قيمة معطاة P1 كدالة متعددة الأسطر فى السطور 40 إلى 120 ويجب أن نبرز أن هذا البرنامج كان يمكن كتابته بكل سهولة وبدون استخدام دالة معرفة بواسطة المبرمج . وفى هذه الحالة ، فان الغرض من الدالة هو هيكله جزء مستقل نسبياً من خطة الحسابات كعنصر متساو من برنامج مستقل . ومن ثم فان ميزة الدالة ميزة تنظيمية أكثر منها حسابية . ومع ذلك وفى بعض البرامج فان وجود الدالة المعرفة بواسطة المبرمج سوف تخلصه من تكرار برمجة نفس مجموعة الحسابات .

```
10 REM          PIGLATIN GENERATOR
20 DIM L(72)
30
40 DEF FNP(P1)
50 REM          THIS FUNCTION FINDS THE END OF A SINGLE WORD
60 FOR I=P1+1 TO 72
70    IF L(I)=ASC( ) THEN 110
80 NEXT I
90 LET FNP = 72
100 GOTO 120
110 LET FNP=I-1
120 FNEND
130
140 REM         READ A LINE OF TEXT
150
160 PRINT
170 PRINT
180 INPUT N$
190 IF N$="END" THEN 410
200 FOR I=1 TO 72
210    LET L(I)=ASC( )
220 NEXT I
230 CHANGE N$ TO L
240 LET P1=1
250 LET P2=FNP(P1)
260
270 REM         PRINT WORD IN PIGLATIN
280
290 IF P2=P1 THEN 330
300 FOR I=P1+1 TO P2
310    PRINT CHR$(L(I));
320 NEXT I
330 PRINT CHR$(L(P1));"A ";
340
350 REM         FIND NEXT WORD
360
370 LET P1=P2+2
380 IF P1>72 THEN 140            END OF LINE
390 IF L(P1)=ASC( ) THEN 140     END OF TEXT
400 GOTO 250
410 END
```

شكل ٦ — ٩

سوف ترى فى شكل ٦ — ١٠ المخرجات المولدة بواسطة البرنامج لثلاثة أسطر نموذجية ، كل سطر يمثل رسالة إدخال يتبعه فورا السطر المناظر من بيجلاتين . وسوف يتوقف تنفيذ البرنامج بعد إدخال الكلمة END من النهاية الطرفية (لاحظ أن اجابات المستخدم موضوع تحتها خط) .

```
?THIS IS A PIGLATIN GENERATOR
HISTA SIA AA IGLATINPA ENERATORGA

?WHAT SORT OF GARBLED MESSAGE IS THIS ANYHOW
HATWA ORTSA FOA ARBLEDGA ESSAGEMA SIA HISTA NYHOWAA

?NOW IS THE TIME FOR ALL GOOD MEN TO COME TO THE AID OF THEIR COUNTRY
OWNA SIA HETA IMETA ORFA LLAA OODGA ENMA OTA OMECA OTA HETA IDAA FOA HEI
RTA OUNTRYCA

?END
```

شكل ٦ — ١٠

٦ — ٦ توليــد أرقام عشــوائية — دالة RND

GENERATING RANDOM NUMBERS—THE RND FUNCTION

هناك عدة تطبيقات شيقة للحاسب مبنية على أساس توليد أرقام عشوائية . فى البيسك يكون من اليسير جداً توليد الأرقام العشوائية بواسطة الدالة المكتبية RND . وترجع الدالة أرقاماً عشوائية مختلفة بقيم تتراوح مابين الصفر والواحد الصحيح ، فى كل مرة يرجع فيها للدالة . ولاتتطلب أى خلاصة .

مثال ٦ — ١٦

يحتوى برنامج بيسك الجمل التالية :

```
20 DIM X(100)
   ...
50 FOR I=1 TO 100
60     LET X(I)=RND
70 NEXT I
```

سوف تتسبب هذه الجملة فى توليد 100 رقم عشوائى وتخزينها فى قائمة X . كل رقم عشوائى سوف يكون كية عشرية قيمتها تقع مابين الصفر والواحد الصحيح .

مثال ٦ — ١٧

نفرض أننا نرغب فى توليد أرقام عشوائية تتراوح قيمها ما بين 3 و 7 يمكن إنجاز ذلك بكتابة :

```
100 LET X=3+(7-3)*RND
```

أو ببساطة

```
100 LET X=3+4*RND
```

مثال ٦ — ١٨

الجملة المبينة فيما بعد سوف تولد رقماً عشوائياً قيمته صحيحة وتقع ما بين 1 إلى 6 سوف تحدث جميع الأرقام بأرجحية متساوية .

```
100 LET X=1+INT(6*RND)
```

وبتفسير هذه الجملة . يجب أن يكون مفهوماً أن دالة RND يمكن أن ترجع قيمة قريبة جداً من الواحد ولكن ليست واحداً . وبذلك

فإن RND ترجع قيمة 0.99999999 ثم 6*RND سوف تنتج قيمة 5.9999994 و INT(6*RND) سوف تنتج قيمة 5 . حيث تحدد قيمة X إذن بالرقم 6 .

سوف نرى برنامج بيسك كاملا يستخدم دالة RND فى مثال ٦ – ٢٠ .

٦ – ٧ جملة RANDOMIZE THE RANDOMIZE STATEMENT

الأرقام التى نحصل عليها من دالة RND ليست عشوائية تماماً ، حيث أنها تولد باستخدام إجراء حسابات منتظمة . ومع ذلك ، تظهر مثل هذه الأرقام كأنها عشوائية ولها نفس الخواص الإحصائية كالأرقام العشوائية الحقيقية . ولذلك فدائماً مايشار إلى هذه الأرقام بالإرقام **العشوائية الكاذبة .**

وفى كل مرة ينفذ فيها برنامج يحتوى على دالة RND سوف تولد نفس الأرقام العشوائية – الكاذبة وبنفس التسلسل . هذا التوالد للأرقام العشوائية يكون عاملا مساعداً عند تتبع البرنامج لاكتشاف الخطأ . ومن الناحية الأخرى ، فغالباً مايكون المطلوب هو توليد تسلسل مختلف من الأرقام العشوائية الكاذبة فى كل مرة ينفذ فيها برنامج التتبع . ويمكن إنجاز ذلك بواسطة **جملة *RANDOMIZE*** .

تتكون جملة RANDOMIZE ببساطة من رقم السطر تتبعه الكلمة الدالة RANDOMIZE . والغرض منها هو إعطاء نقطة بداية مختلفة لمولد الأرقام العشوائية . وبذلك فيجب أن تسبق جملة RANDOMIZE المرجع الأول للدالة RND فى البرنامج .

مثال ٦ – ١٩

برنامج بيسك يحتوى على الجمل :

```
20 DIM X(100)
30 RANDOMIZE
   ...
50 FOR I=1 TO 100
60      LET X(I)=RND
70 NEXT I
```

سوف تسبب هذه الجمل فى توليد 100 رقم عشوائى كاذب وتخزينها فى القائمة X ، كما فى المثال ٦ – ١٦ . بينما يختلف عن مثال ٦ – ١٦ ، فى مجموعة من الأرقام العشوائية المختلفة سوف يتم توليدها فى كل مرة ينفذ فيها البرنامج .

مثال ٦ – ٢٠ لعبة حظ (تصويب كرابس) A Game of Chance (Shooting Craps)

فى هذا المثال سوف نحاكى لعبة كرابس على الحاسب . والكرابس لعبة زهر شعبية حيث يرمى اللاعب زوجاً من النرد (الزهر) مرة أو أكثر حتى يكسب أو يخسر . ويمكن تنفيذ اللعبة على الحاسب بالتعويض بالأرقام العشوائية المولدة عن الرمى الفعلى للزهر .

قواعد اللعبة Rules of the Game

توجد طريقتان يمكن بهما أن يكسب اللاعب فى لعبة الكرابس . يمكنه رمى الزهر مرة ويحصل فيها على 7 أو 11 نقطة أو يمكن أن يحصل على 4 أو 5 أو 6 أو 8 أو 9 أو 10 من الرمية الأولى ، ثم يحصل على نفس النقاط فى رميات متعاقبة قبل الحصول على 7 نقاط . وعلى النقيض فهناك طريقتان يمكن أن يخسر بهما اللاعب يمكنه إما أن يرمى الزهر مرة ويحصل على نقطتين أو 3 أو 12 أو يحصل على 4 نقاط أو 5 أو 6 أو 8 أو 9 أو 10 فى الرمية الأولى ، ثم يحصل على 7 نقاط فى الرمية التالية قبل الرجوع بنفس عدد النقاط التى حصل عليها فى الرمية الأولى .

طريقة إجراء الحسابات Computational Procedure

دعنا نبرمج اللعبة بطريقة تخاطبية ، حيث تحاكي رمية الزهر في كل مرة يضغط اللاعب لرجوع العربة على النهاية الطرفية . بعد ذلك تظهر رسالة مشيرة إلى نتيجة كل رمية . وسوف تستمر اللعبة حتى يطبع اللاعب كلمة END وبالإضافة إلى ذلك ، فسوف نضمن طباعة قواعد اللعبة إذا ما أدخل اللاعب الكلمة RULES .

من أجل محاكاة رمية واحدة للزهر سوف نولد رقمين عشوائيين كل منها يحتوي على رقم صحيح يقع ما بين 1 إلى 6 مجموع هذين العددين سوف يمثل الهدف الذي تم الحصول عليه عند رمية الزهر . ومن المناسب أن تستخدم دالة معرفة بواسطة المبرمج لهذا الغرض حيث أن كل مرة يتم فيها الإشارة إلى الدالة سوف تتم محاكاة رمية أخرى للزهر .

الجزء الرئيسي من البرنامج سوف يفحص الهدف الذي تم الحصول عليه من كل رمية ويحدد ما إذا كان اللاعب من الخاسرين أو الناجحين أو يتطلب رمية أخرى . وفي كل حالة سوف تتم طباعة رسالة ملائمة مع الهدف الذي تمت محاكاته ، وسوف يتضمن أيضاً كتلة من جمل PRINT والتي يترتب عليها طباعة القواعد للإجابة على اللاعب الذي يكتب RULES .

المخطط التمهيدي للبرنامج The Program Outline

١ – استهل مولد الرقم العشوائي .

٢ – إقرأ قيمة للمتغير الحرفي N$.

(أ) أوقف التنفيذ إذا كانت N$=END

(ب) إطبع قواعد اللعبة إذا كانت N$=RULES ثم كرر هذه الخطوة (أي ، أقرأ قيمة جديدة للمتغير N$)

(ﺣ) تقدم إلى الخطوة ٣ إذا كانت N$ تمثل سلاسل حرفية غير END أو RULES . (لاحظ أن أي حرف على حدة يمكن أن يعطى للمتغير N$ لهذا الغرض . يمكن أن يكون أكثرها ملاءمة لهذا الغرض قضيب ترجيع العربة ، وذلك لوضوحه على الآلة الكاتبة الطرفية) .

٣ – قم بمحاكاة رمي الزهر مرة واحدة ، مستدعياً نتيجة الهدف (K) .

(أ) إذا كانت قيمة K هي 2 أو 3 أو 12 فذلك يشير إلى الخسارة ثم بعد ذلك اطبع رسالة ملائمة وارجع إلى الخطوة ٢ مرة أخرى .

(ب) إذا كانت قيمة K هي 7 أو 11 فذلك يشير إلى المكسب . ثم بعد ذلك اطبع رسالة ملائمة وارجع إلى الخطوة ٢ مرة أخرى.

(ﺣ) إذا كانت K قيمتها 4 أو 5 أو 6 أو 8 أو 9 أو 10 فإن ذلك يتطلب رمية إضافية للزهر . ثم اطبع رسالة ملائمة وتقدم إلى الخطوة رقم ٤ .

٤ – اجعل K1 = K (سوف يسمح بمقارنة القيم المتعاقبة للمتغير K أن تقارن بالقيمة الأصلية ، حيث تسمى الآن K1) .

٥ – إقرأ قيمة جديدة للمتغير N$ ثم قم بمحاكاة رمية أخرى للزهر ، وبذلك تولد قيمة جديدة للمتغير K (لاحظ أن الزهر لن يرمى حتى يعطي اللاعب إشارة بإدخال بعض الحروف ، ومن أمثال هذه الحروف إرجاع العربة من أجل N$) .

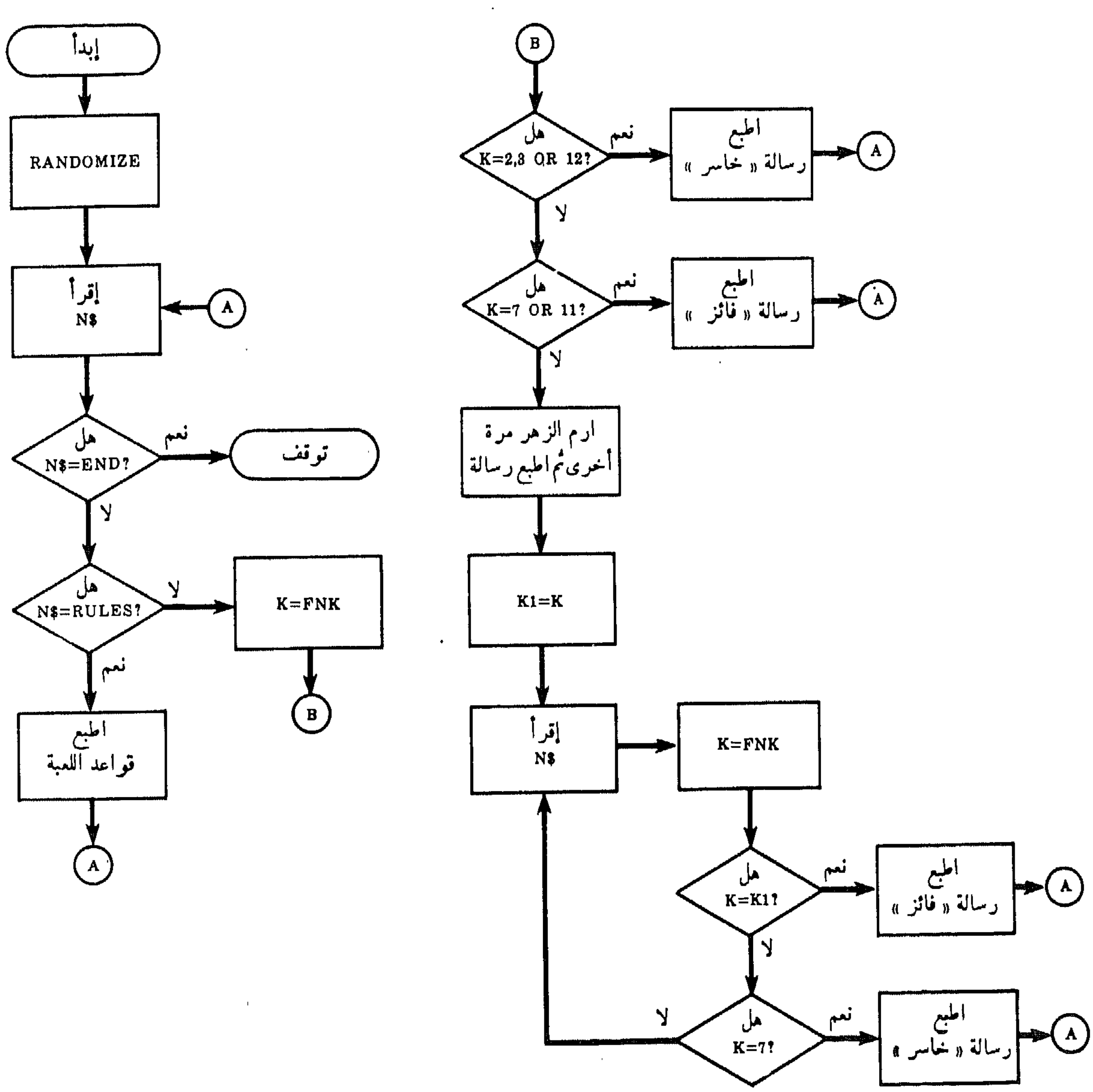
٦ – قارن K بالمتغير K1 .

(أ) إذا كانت K1 = K فإن ذلك يشير إلى المكسب . من ثم اطبع رسالة ملائمة ثم ارجع للخطوة رقم ٢ .

(ب) إذا كانت قيمة K = 7 فإن ذلك يشير إلى المكسب . إذن اطبع رسالة ملائمة ثم ارجع للخطوة رقم ٢ .

(ﺣ) إذا كانت K لاتساوى K1 ولا تساوى 7 ، إذن ارجع لخطوة رقم ٥ وقم بتوليد قيمة جديدة للمتغير K .

خريطة لسير عمليات هذا الإجراء مبينة فى شكل ٦ — ١١ .



شكل ٦ — ١١

سوف نقوم بعملية محاكاة رمية واحدة للزهر فى دالة معرفة بواسطة المبرمج كما يلى :

١ — اجعل $K2 = 1 + INT(6*RND)$

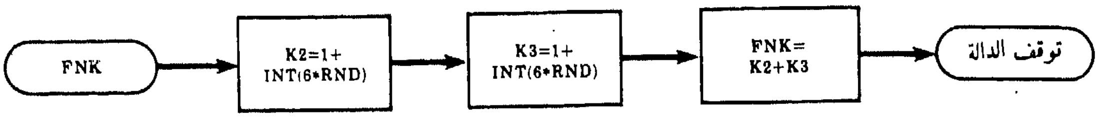
(أنظر المثال ٦ — ١٨ لشرح هذه الجملة)

٢ — اجعل $K3 = 1 + INT(6*RND)$

٣ — اجعل $K = K2 + K3$

٤ – ارجع القيمة الحالية للمتغير K إلى نقطة الإشارة للدالة .

يبين شكل ٦ – ١٢ خريطة سير عمليات مناظرة له . لاحظ أن القيمة المحسوبة للمتغير K تسمى FNK بداخل الدالة .



شكل ٦ – ١٢

برنامج بيسك The BASIC Program

يحتوى شكل ٦ – ١٣ على برنامج كامل للقيام بالحسابات . لاحظ أن السطور من 500 إلى 560 تعرف دالة متعددة السطور FNK حيث تحاكى رمية زوج من الزهر . وقد تم الرجوع إلى هذه الدالة مرتين فى نقطتين مختلفتين فى البرنامج – بالتحديد السطور 90 و 180 .

سوف نرى من خلال الدالة FNK المعرفة بواسطة المبرمج أننا أشرنا إلى الدالة المكتبية RND ولاحظ، أيضاً جملة RANDOMIZE فى السطر رقم 20 ، والذى استخدم ليستهل مولد الرقم العشوائى فى كل مرة ينفذ فيها البرنامج . من المهم أن تسبق جملة RANDOMIZE الإشارة إلى الدالة RND (من خلال الإشارة إلى الدالة FNK)

```
10 REM          SIMULATION OF A GAME OF CRAPS
20 RANDOMIZE
30 DEF FNK
40 LET K2=1+INT(6*RND)
50 LET K3=1+INT(6*RND)
60 LET FNK=K2+K3
70 FNEND
80 INPUT N$
90 IF N$="END" THEN 460
100 IF N$="RULES" THEN 330
110
120 REM          SIMULATE ONE PLAY OF CRAPS
130
140 LET K=FNK
150 ON (K-1) GOTO 160,160,200,200,200,180,200,200,200,180,160
160 PRINT K; "- - YOU LOSE ON THE FIRST THROW"
170 GOTO 300
180 PRINT K; "- - YOU WIN ON THE FIRST THROW"
190 GOTO 300
200 LET K1=K
210 PRINT K; "- - THROW THE DICE AGAIN"
220 INPUT N$
230 LET K=FNK
240 IF K=K1 THEN 270
250 IF K=7 THEN 290
260 GOTO 210
270 PRINT K; "- - YOU WIN BY MATCHING YOUR FIRST SCORE"
280 GOTO 300
290 PRINT K; "- - YOU LOSE BY FAILING TO MATCH YOUR FIRST SCORE"
300 PRINT
310 GOTO 80
320
330 REM          PRINT RULES OF CRAPS
340
350 PRINT,"RULES OF CRAPS"
360 PRINT
370 PRINT "TO WIN: OBTAIN A 7 OR 11 ON THE FIRST THROW"
380 PRINT "     OR: OBTAIN A 4,5,6,8,9 OR 10 ON THE FIRST THROW"
390 PRINT,"AND MATCH YOUR ORIGINAL SCORE BEFORE THROWING A 7"
400 PRINT
410 PRINT "TO LOSE: OBTAIN A 2,3 OR 12 ON THE FIRST THROW"
420 PRINT "      OR: OBTAIN A 4,5,6,8,9 OR 10 ON THE FIRST THROW"
430 PRINT,"AND THROW A 7 BEFORE MATCHING YOUR ORIGINAL SCORE"
440 PRINT
450 GOTO 80
460 END
```

مثال ٦ – ٢١

نبين فيما يلي برنامج فرعى نموذجى :

```
300 REM SUBROUTINE TO CALCULATE CRITICAL CONSTANTS
310 LET C1=(A+B C)/3
320 LET C2=SQR(A↑2+B↑2+C↑2)
330 LET C3=SQR(A*B*C)
340 RETURN
```

لاحظ أن البرنامج الفرعى يبدأ بجملة REM وينتهى بجملة RETURN يجب أن تحدد المتغيرات A و B و C قيما عددية قبل الإشارة إلى البرنامج الفرعى .

يمكن أن يحتوى البرنامج الفرعى الصغير عدة جمل RETURN عند الحاجة وغالباً ما يكون ذلك ضرورياً وله أهمية كبيرة إذا كان البرنامج الفرعى يحتوى على تفرع مشروط أو تفرع متعدد . وعند تنفيذ البرنامج الفرعى فإن أول جملة RETURN يقابلها سوف تسبب تحويل التحكم من البرنامج الفرعى .

مثال ٦ - ٢٢

الهيكل التخطيطى لبرنامج فرعى يحتوى على عدة جمل RETURN مبينة فيما يلى :

```
500 REM SAMPLE SUBROUTINE WITH MULTIPLE RETURNS
510 ON N 520,580,650
520 LET X=...
    ...
-70 RETURN
580 LET Y=...
    ...
640 RETURN
650 LET Z=...
    ...
690 RETURN
```

يحتوى هذا البرنامج الفرعى على تفرعات عديدة حيث يتحول التحكم إلى الجمل 520 و 580 و 650 (وكل ذلك يتضمنه البرنامج الفرعى) متوقفاً فى ذلك على قيمة N . ثم بعد ذلك يتحول التحكم مرة أخرى إلى الجملة التى تلى الإشارة للبرنامج الفرعى عند مقابلة أى جملة من جمل RETURN .

٦ — ٩ الاشارة الى برنامج فرعى — جملة GOSUB

REFERENCING A SUBROUTINE—THE GOSUB STATEMENT

يمكن الإشارة إلى البرنامج الفرعى بواسطة **جملة** *GOSUB* تتكون هذه الجملة من رقم للسطر ، ثم الكلمة الدالة GOSUB ثم بعد ذلك رقم أول جملة فى البرنامج الفرعى . وعند تنفيذ هذه الجملة فإنها تحول التحكم إلى البرنامج الفرعى . وسوف يتحول التحكم مرة أخرى إلى الجملة التى تلى GOSUB عند مقابلة جملة RETURN بداخل البرنامج الفرعى .

مثال ٦ – ٢٣

برنامج بيسك يحتوى على الجمل التالية :

```
120 GOSUB 300
130 PRINT "Z=";Z
    ...
300 LET X=A+B  }
    ...        }  برنامج فرعى صغير
380 RETURN     }
```

فى شكل ٦ – ١٤ نرى قائمة تمثل مخرجات البيانات . أولاً نرى أن القواعد قد تمت طباعتها عند طلب اللاعب لذلك بكتابة RULES . يلى ذلك ٥ ألعاب نمطية (٣ مكسب و ٢ خسارة) . وأخيراً كتب اللاعب NED ، ويترتب على ذلك أنهاء تنفيذ البرنامج . (لاحظ أن استجابه المستخدم تحتها خط .

```
?RULES
          RULES OF CRAPS


TO WIN: OBTAIN A 7 OR 11 ON THE FIRST THROW
    OR: OBTAIN A 4,5,6,8,9 OR 10 ON THE FIRST THROW
            AND MATCH YOUR ORIGINAL SCORE BEFORE THROWING A 7


TO LOSE: OBTAIN A 2,3 OR 12 ON THE FIRST THROW
     OR: OBTAIN A 4,5,6,8,9 OR 10 ON THE FIRST THROW
            AND THROW A 7 BEFORE MATCHING YOUR ORIGINAL SCORE

?
11 - - YOU WIN ON THE FIRST THROW

?
5 - - THROW THE DICE AGAIN
?
7 - - YOU LOSE BY FAILING TO MATCH YOUR FIRST SCORE

?
3 - - YOU LOSE ON THE FIRST THROW

?
7 - - YOU WIN ON THE FIRST THROW

?
4 - - THROW THE DICE AGAIN
?
8 - - THROW THE DICE AGAIN
?
9 - - THROW THE DICE AGAIN
?
11 - - THROW THE DICE AGAIN
?
6 - - THROW THE DICE AGAIN
?
5 - - THROW THE DICE AGAIN
?
4 - - YOU WIN BY MATCHING YOUR FIRST SCORE

?END
```

شكل ٦ – ١٤

٦ — ٨ تعريف برنامج فرعى DEFINING A SUBROUTINE

فى بعض الأحيان يكون من الأكثر ملاءمة بناء هياكل من الجمل المتتالية كبرنامج فرعى بدلا من دالة . البرامج الفرعية مشابهة للدول بمعنى أننا يمكن الرجوع إليها من أماكن أخرى من البرنامج . يختلف البرنامج الفرعى عن الدالة من حيث أنه ليس له إسم كما أن البرنامج الفرعى يمكن أن يستخدم ليحدد قيما لأكثر من كمية عددية و / أو سلاسل حرفية . وعلاوة على ذلك لانستخدم الخلاصات . ومن ثم فإن البرنامج الفرعى يتبادل المعلومات مع باقى البرنامج بطريقة عامة جداً .

لا يستلـزم أن نبـدأ البرنامـج الفرعى بجملـة خاصـة . وعلى ذلـك يمكننـا بداية البرنامـج الفرعى بجملـة REM أو جملـة LET أو جملة FOR – TO أو جملة إدخال . . . . الخ) . ولكن يجب أن يكون آخر جملة هى جملة *RETURN* حيث تتكون ببساطة من رقم جملة تتبعه الكلمة الدالة RETURN وتسبب هذه الجملة تحويل التحكم مرة أخرى إلى الجملة الموجودة التى تلى نقطة الإشارة إلـى البرنامـج الفرعى . (يجب أن يفهم أن التحكم لايمكن أن يتحول بواسطة أى جملة من جمل التفرع ، كثل GO TO أو IF-THEN أو ON-GO TO) .

عند مقابلة الجملة رقم 120 (GOSUB) أثناء تنفيذ البرنامج . سوف يتحول التحكم إلى الجملة رقم 300 وسوف ينفذ البرنامج الفرعى . وعند الوصول إلى الجملة رقم 380 (RETURN) فسوف يتحول التحكم مرة أخرى إلى الجملة رقم 130 وهى أول جملة بعد GOSUB .

يمكن أن يحتوى البرنامج على أكثر من استدعاء لنفس البرنامج الفرعى . ودائماً يتحول التحكم مرة أخرى من البرنامج الفرعى إلى الجملة التى تلى جملة GOSUB المعنية التى أشارت إلى البرنامج الفرعى .

مثال ٦ – ٢٤

برنامج بيسك يحتوى على الجمل التالية :

```
120 GOSUB 300
130 PRINT "Z=";Z
    ...
180 GOSUB 300
190 IF Z<10 THEN 250
    ...
300 LET X=A+B  }
    ...        }  برنامج فرعى صغير
380 RETURN     }
```

تعرف الجمل 300 إلى 380 البرنامج الفرعى ، كما فى مثال ٦ – ٢٣ . إذا استدعى البرنامج الفرعى بواسطة الجملة رقم 120 ، فإن التحكم سوف يعود مرة أخرى إلى الجملة رقم 130 بعد إتمام تنفيذ البرنامج الفرعى . وبالمثل ، إذا استدعى البرنامج الفرعى بواسطة الجملة رقم 180 ، فإن التحكم سوف يعود مرة أخرى إلى الجملة رقم 190 بعد إتمام تنفيذ البرنامج الفرعى .

من الممكن أن يستدعى برنامج فرعى برنامجاً فرعياً آخر . والبرامج الفرعية التى تم هيكلتها بهذه الطريقة تسمى برامج فرعية **متداخلة** (تذكر أننا قابلنا هذا المصطلح فى الفصل الرابع ، عند مناقشة الحلقات التكرارية المتداخلة FOR-TO) .

مثال ٦ – ٢٥

برنامج بيسك يحتوى على الجمل التالية :

```
 50 GOSUB 200
    ...
200 LET C=A+B  }
    ...        }
240 GOSUB 300  }  البرنامج الفرعى الأول
    ...        }
270 RETURN     }
    ...
300 LET P=Q+R  }
    ...        }  البرنامج الفرعى الثانى
350 RETURN     }
```

يحتوى هذا البرنامج على برنامجين فرعيين . يتكون البرنامج الفرعى الأول من الجمل 200 حتى 270 ، ويتكون البرنامج الفرعى الثانى من الجمل 300 حتى 350 لاحظ أننا استدعينا البرنامج الفرعى الثانى من نقطة بداخل البرنامج الفرعى الأول (السطر 240) ، ومن ثم فإن البرنامجين الفرعيين متداخلين .

عند مقابلة الجملة رقم 350 أثناء تنفيذ البرنامج ، فإن التحكم سوف يتحول مرة ثانية إلى الجملة التى تلى السطر 240 . وبذلك فإن التحكم تحول من البرنامج الفرعى الثانى إلى البرنامج الفرعى الأول . وبالمثل فإن الجملة رقم 270 سوف تحول التحكم ثانية إلى الجملة التى تلى السطر رقم 50 ، وبذلك يتم تحويل التحكم من البرنامج الفرعى الأول إلى نقطة الاستدعاء الأولى .

يجب أن يحتفظ بالبرامج الفرعية المتداخلة على ترتيب هرمى دقيق . وهو ، إذا استدعى البرنامج الفرعى A البرنامج الفرعى B ، فإن البرنامج الفرعى B لا يمكنه استدعاء البرنامج الفرعى A . وعلى الوجه الآخر ، فإن البرنامج الفرعى B يمكن استدعاءه ومن الجزء الأساسى فى البرنامج بجانب استدعاءه من البرنامج الفرعى A .

توضح الأمثلة ٦ – ٢٦ و ٦ – ٢٨ استخدام البرامج الفرعية فى برامج بيسك كاملة .

مثال ٦ – ٢٦ المرتبـات الشهرية A Monthly Payroll

فى هذا المثال سوف يتم تحديد قيمة ضريبة الدخل الفيدرالية وضريبة الولاية والضريبة المحلية التى يجب استقطاعها من المرتب الأساسى للموظف طبقاً لحالته الاجتماعية وعدد المعولين. ولن نحاول إجراء حسابات كاملة للمرتبات ، لأن هيكل البرنامج سيكون معقداً إلى حد ما .

يبين جدول ٦ – ٢ مقدار ضريبة الدخل الفيدرالية التى يجب أن تستقطع على أساس شهرى لكل من الموظف الأعزب والمتزوج .

جدول ٦ – ٢ نسبة استقطاعات ضريبة الدخل الفيدرالية

فترة المرتبات الشهرية							
شخص أعـزب – متضمنا – رب الأسرة				شخص متزوج			
اذا كان مبلغ الأجـور هو :		قيمة ضريبة الدخل التى سـوف تستقطع هى :		اذا كان مبلغ الأجـور هو :		مقدار ضريبة الدخل التى سـوف تستقطع هى :	
لا يزيد عن $88		0		لا تزيد عن $88		0	
أكبر من	ولكن لايزيد عن		مايزيد عن	اكبر من	ولكن لايزيد عن		ما يزيد على
$88	–$133	14%	–$88	$88	–$183	14%	–$88
$133	–$217	$6.30 plus 17%	–$133	$183	–$333	$13.30 plus 17%	–$183
$217	–$433	$20.58 plus 20%	–$217	$333	–$708	$38.80 plus 16%	–$333
$433	–$583	$63.78 plus 18%	–$433	$708	–$1167	$98.80 plus 19%	–$708
$583	–$917	$90.78 plus 21%	–$583	$1167	–$1667	$186.01 plus 21%	–$1167
$917	—	$160.92 plus 24%	–$917	$1667	—	$291.01 plus 25%	–$1667

استخدام هذه الجداول يجب أن يحسب الدخل الأساسى الشهرى المعدل ، وهو مساو للدخل الأساسى الشهرى مطروحاً منه 54.20 $ لكل معول . نسبة الضريبة المطلوبة سوف تعتمد على شريحة ضريبة الدخل الأساسى التى يقع بداخلها الموظف . لاحظ أنه توجد مجموعتان مختلفتان لشرائح الضريبة – احدهما للموظف الأعزب والأخرى للمتزوج .

سوف تحسب الضريبة لحساب الولاية بنسبة واحد فى المائة من اجمالى الدخل حتى 600 $ فى الشهر ، و 1½ فى المائة من الدخل الإضافى حتى 2000 $ فى الشهر ، و 2 فى المائة لما يزيد عن 2000 $. وسوف تحصل الضريبة المحلية بنسبة واحد فى المائة من إجمالى الدخل الشهرى الذى يقل عن 800 $ ولن يخضع الدخل الشهرى الذى يزيد عن 800 $ للضرائب المحلية .

طريقة إجـراء الحسابات Computational Procedure

دعنا نحسب الدخل الإجمالى الشهرى المعدل ، وضريبة الولاية والضريبة المحلية فى برنامج فرعى ، ثم الضريبة الفيدرالية فى برنامج فرعى آخر . ويسمح لنا ذلك بفصل البرنامج إلى عدة أجزاء منفصلة . وسوف تكون البرامج الفرعية متداخلة وذلك باستدعاء البرنامج الفرعى للضريبة الفيدرالية بواسطة البرنامج الفرعى الآخر . أما باقى البرنامج فسوف يقرأ البيانات المطلوبة ببساطة (وهى اسم الموظف ورقمه والمرتب الإجمالى الشهرى والحالة الاجتماعية وعدد المعولين) ثم يستدعى البرامج الفرعية ، وبعد ذلك يحسب صافى المرتب الشهرى وأخيراً يطبع النتائج المطلوبة (وهى : مقدار الضريبة الفيدرالية وضريبة الولاية والضريبة المحلية التى يجب استقطاعها وصافى المرتب الشهرى) لكل موظف .

وسوف نشير من داخل البرنامج الفرعى للضريبة الفيدرالية إلى الأرقام الموجودة فى أعمدة الطرف الأيسر من جدول الضرائب ، والذى يقرر حدود الدخل العديدة ، كعناصر للمجموعة المتراصة C وسوف نعرف C على أنها مجموعة متراصة ذات بعدين (2 × 6) إذا كان الدليل الثانى J يساوى 1 ، سوف تشير إلى بيانات الرجل الأعزب ، بينما عندما J = 2 تشير إلى بيانات الرجل المتزوج ومن ثم . C(1,1) = 88 و C (2,1) = 133 و . . . و C (6,1) = 917 و C (1, 2) = 88 و C (2, 2) = 183 و . . . و C (6, 2) = 1667

وبطريقة مماثلة اجعل T مجموعة متراصة ذات بعدين (2 × 6) تحتوى على قيمة أساس حدود ضريبة الدخل ، R مجموعة متراصة ذات بعدين (2 × 6) تمثل نسبة الضريبة (كرقم عشرى) لكل من حدود الدخل . وبذلك فإن T (1,1) = 0 و T (2, 1) = 6.30 و T (6, 1) = 160.92 و T (1, 2) = 0 و T(2, 2) = 13.30 و .... T(6, 1) = 291.01, R(1, 1) = 0.14, و R(2, 1) = 0.17 و .....R(6, 1) = 0.24 و R(1, 2) = 0.14 و R(2, 2) = 0.17, و .... R(6, 2) = 0.25,

المخطط التمهيدى للبرنـــامج The Program Outline

دعنا نعرف الرموز التالية :

M$ = اسم الموظف
N$ = الحالة الاجتماعية (M للمتزوج و S للأعزب)
N = رقم الموظف
E = عدد المعولين
P1 = المرتب الأساسى الشهرى
P2 = المرتب الإجمالى الشهرى المعدل .
P3 = صافى المرتب الشهرى
T1 = الضريبة المستقطعة للولاية
T2 = الضريبة المحلية المستقطعة
T3 = الضريبة الفيدرالية المستقطعة

وسوف تتم العمليات الحسابية كما هو مبين فيما يلى :

١ – حدد قيما رقمية للمجموعات المتراصة C و R و T ذات البعد (2 × 6) وذلك بواسطة جملتى READ و DATA (حلقتين تكراريتين من حلقات FOR-TO سوف تكون مطلوبة لكل مجموعة متراصة ، كما ناقشنا ذلك فى الفصل الخامس) .

٢ – إقرأ قيمة حرفية مناسبة للمتغير N$

(أ) إذا كانت N$=END إنه الحسابات

(ب) وإلا فواصل لخطوة رقم ٣ التالية

٣ – إقرأ قيما رقمية وحرفية لكل من N و P1 و M$ و E

٤ – احسب قيما رقمية لكل من T1 و T2 و T3 وذلك باستدعاء البرامج الفرعية المناسبة .

٥ – احسب قيمة P3 باستخدام الصيغة

$$P3 = P1 - (T1 + T2 + T3)$$

٦ – اطبع القيم الحالية لكل من T3 و T1 و T2 و P3 .

٧ – ارجع لخطوة رقم ٢ وابدأ معالجة بيانات الموظف التالى :

وسوف نواصل البرنامج الفرعى الذى يشار إليه مباشرة فى الخطوة رقم ٤ كما يلى :

١ – افحص قيمة P1

(أ) إذا لم تزد P1 عن 600 $ ، فاحسب قيمة T1 (ضريبة الولاية)

$$T1 = 0.01*P1$$

ثم واصل إلى الخطوة رقم ٢ أدناه .

(ب) إذا زادت قيمة P1 عن 600 $ ولكن لم تتعد 2000 $ ، فاحسب قيمة T1 باستخدام الصيغة :

$$T1 = 6 + 0.015*(P1 - 600)$$

ثم واصل إلى الخطوة رقم ٢ أدناه .

(حـ) أما إذا زادت قيمة P1 عن 2000 $ ، فاحسب T1

$$T1 = 27 + 0.02*(P1 - 2000)$$

ثم واصل إلى الخطوة رقم ٢ أدناه .

٢ – ومرة أخرى افحص قيمة P1

(أ) إذا لم تزد قيمة P1 عن 800 $ ، احسب قيمة T2 (الضريبة المحلية)

$$T2 = 0.01*P1$$

ثم واصل إلى الخطوة رقم ٣ أدناه .

(حـ) إذا زادت قيمة P1 عن 800$ اجعل قيمة T2 = 8 ثم واصل لخطوة رقم ٣ أدناه .

٣ – احسب قيمة P2 (المرتب الشهرى الأساسى المعدل) باستخدام الصيغة

$$P2 = P1 - 54.20*E$$

(أ) إذا كانت P2 أقل أو تساوى صفراً ، اجعل قيمة T3 (الضريبة الفيدرالية) تساوى صفراً ثم ارجع مرة أخرى إلى الجزء الأساسى من البرنامج .

(ب) أما إذا كانت P2 لها قيمة موجبة إذن ارجع للبرنامج الفرعى الذى يقرر قيمة T3 ثم ارجع مرة أخرى لجزء الأساسى من البرنامج .

وسوف يواصل البرنامج الفرعى الذى يحسب قيمة T3 (الضريبة الفيدرالية) بالطريقة التالية :

١ – قرر ما إذا كان الموظف أعزب أو متزوجاً وذلك بفحص السلسلة الحرفية M$.

(أ) إذا كانت M$=S ، اعط قيمة 1 إلى J ثم واصل للخطوة رقم ٢

(ب) إذا كانت M$=M ، اجعل قيمة J = 2 ثم واصل لخطوة ٢ أدناه .

٢ – اجعل قيمة T3 المبدئية مساوية للصفر .

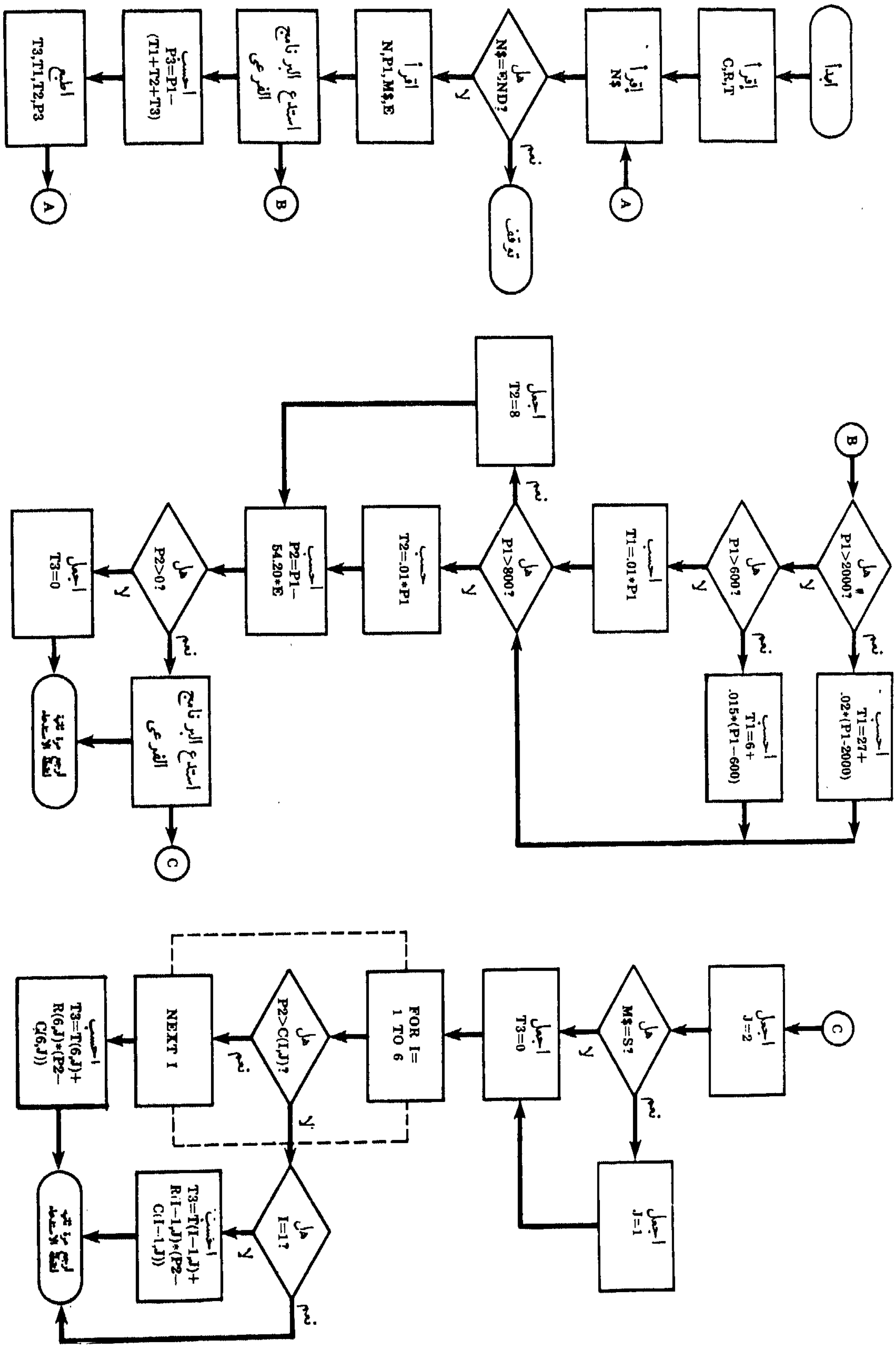
٣ – قارن قيمة P2 بالعنصر C(I, J) لكل قيم I التى تتراوح من 1 إلى 6

(أ) إذا كانت P2 لاتتعدى قيمة C (1, J) احتفظ بقيمة T3 = 0 ثم ارجع إلى نقطة استدعاء البرنامج الفرعى .

(ب) إذا تعدت قيمة P2 قيمة C (1, J) واصل خلال الحلقة التكرارية حتى نصل إلى قيمة C (I, J) التى تتعدى P2 . إذا وجدت مثل هذه القيمة فاحسب قيمة T3 .

```
T3=T(I-1,J)+R(I-1,J)*(P2-C(I-1,J))
```

ثم ارجع إلى نقطة استدعاء البرنامج الفرعى .


ابدأ
اقرأ C,R,T
اقرأ N$
هل N$=END?
نعم
توقف
لا
اقرأ N,P1,M$,E
استدع البرنامج الفرعي
احسب P3=P1−(T1+T2+T3)
اطبع T3,T1,T2,P3
هل P1>2000?
احسب T1=27+.02*(P1-2000)
هل P1>600?
احسب T1=6+.015*(P1−600)
احسب T1=.01*P1
هل P1>800?
اجعل T2=8
احسب T2=.01*P1
احسب P2=P1−54.20*E
هل P2>0?
اجعل T3=0
ارجع إلى البرنامج
اجعل J=2
هل M$=S?
اجعل J=1
اجعل T3=0
FOR I= 1 TO 6
هل P2>C(I,J)?
NEXT I
هل I=1?

شكل ٦ - ١٠

(ﺟ) إذا تعدت قيمة P2 قيمة C (I, J) عند نهاية الحلقة التكرارية (أى عندما تكون I = 6) ، إذن احسب قيمة T3 باستخدام الصيغة

$$T3=T(6,J)+R(6,J)*(P2-C(6,J))$$

ثم ارجع مرة أخرى إلى نقطة استدعاء البرنامج الفرعى .

يحتوى شكل ٦ – ١٥ على خريطة سير عمليات مفصلة تناظر المخطط التمهيدى السابق .

```
10 REM        COMPUTATION OF A MONTHLY PAYROLL
20 DIM C(6,2),R(6,2),T(6,2)
30 FOR J=1 TO 2
40    FOR I=1 TO 6
50       READ C(I,J)
60    NEXT I
70 NEXT J
80 FOR J=1 TO 2
90    FOR I=1 TO 6
100      READ R(I,J)
110   NEXT I
120 NEXT J
130 FOR J=1 TO 2
140   FOR I=1 TO 6
150      READ T(I,J)
160   NEXT I
170 NEXT J
180 PRINT,"MONTHLY PAYROLL"
190 PRINT
200 PRINT "NAME";
210 INPUT N$
220 IF N$="END" THEN 800
230 PRINT "EMPLOYEE NUMBER";
240 INPUT N
250 PRINT "GROSS SALARY";
260 INPUT P1
270 PRINT "MARITAL STATUS (M OR S)";
280 INPUT M$
290 PRINT "NUMBER OF EXEMPTIONS";
300 INPUT E
310 GOSUB 400
320 LET P3=P1-(T1+T2+T3)
330 PRINT "FEDERAL TAX=$";T3,"STATE TAX=$";T1,"LOCAL TAX=$";T2
340 PRINT "NET SALARY=$";P3
350 GOTO 190
360 DATA 88,133,217,433,583,917,88,183,333,708,1167,1667
370 DATA .14,.17,.20,.18,.21,.24,.14,.17,.16,.19,.21,.25
380 DATA 0,6.3,20.58,63.78,90.78,160.92,0,13.3,38.8,98.8,186.01,291.01
390
400 REM COMPUTATION OF STATE TAX, LOCAL TAX AND ADJUSTED GROSS SALARY
410
420 IF P1>2000 THEN 480
430 IF P1>600 THEN 460
440 LET T1=.01*P1
450 GOTO 490
460 LET T1=6+.015*(P1-600)
470 GOTO 490
480 LET T1=27+.02*(P1-2000)
490 IF P1>800 THEN 520
500 LET T2=.01*P1
510 GOTO 530
520 LET T2=8
530 LET P2=P1-54.2*E
540 IF P2>0 THEN 570
550 LET T3=0
560 RETURN
570 GOSUB 600
580 RETURN
590
600 REM -       COMPUTATION OF FEDERAL TAX
610
620 LET J=2
630 IF M$="M" THEN 650
640 LET J=1
650 LET T3=0
660 FOR I=1 TO 6
670    IF P2>C(I,J) THEN 710
680    IF I=1 THEN 730
690    LET T3=T(I-1,J)+R(I-1,J)*(P2-C(I-1,J))
700    GOTO 730
710 NEXT I
720 LET T3=T(I,J)+R(I,J)*(P2-C(I,J))
730 RETURN
800 END
```

شكل ٦ – ١٦

برنـــامج بيســـك The BASIC Program

يبين شكل ٦ – ١٦ برنامج بيسك كاملا . لاحظ أول برنامج فرعى ، والذى يحسب الدخل الأساسى المعدل وضريبة الولاية والضريبة المحلية ويتكون من الجمل 400 إلى 580 ، ويتضمن جملتى RETURN (الأسطر 560 و 580) . ويستدعى البرنامج الفرعى بواسطة الجملة رقم 310 فى الجزء الأساسى من البرنامج .

وتكون الجمل 600 إلى 730 البرنامج الفرعى الثانى ، والذى يستخدم لحساب الضريبة الفيدرالية . ونرى أن هذا البرنامج الفرعى لايحتوى إلا على جملة RETURN واحدة (السطر 730) ويستدعى هذا البرنامج الفرعى فى السطر رقم 570 . (من ثم فإن البرامج الفرعية متداخلة) . لاحظ أن حلقة FOR-TO التكرارية (السطور من 660 إلى 710) متضمنة فى البرنامج الفرعى الثانى .

ومن الممتع أيضاً ملاحظة أن البرنامج يحتوى على جمل READ و INPUT ، وتستخدم READ لإعطاء قيم للمجموعات المتراصة C و R و T عند بداية تنفيذ البرنامج . بينما جمل INPUT تستخدم لإدخال المعلومات المطلوبة لكل موظف .

يجب أن يفهم القارىء أن هذا البرنامج كان يمكن كتابته بسهولة وبدون استخدام البرامج الفرعية . ومن ثم فالغرض من استخدام البرامج الفرعية فى هذا المثال هو تنظيم البرنامج على هيئة عدة أجزاء معرفة تماماً ومستقلة ، ولقد رأينا موقفاً مماثلا بالنسبة للدوال المعرفة بواسطة المبرمج فى مثال ٦ – ١٥ . بينما ، توجد مواقف عديدة يمكن فيها تبسيط البرمجة بطريقة ملحوظة بواسطة استخدام البرامج الفرعية . ويكون هذا صحيحاً خاصة فى البرامج التى تحتوى على عدة جمل لاستدعاء نفس البرنامج الفرعى .

وأخيراً ، يبين شكل ٦ – ١٧ مجموعة مخرجات نموذجية لعدد 7 موظفين مختلفين . لاحظ أن كل البيانات (سواء مدخلة أو مخرجة) الخاصة بكل موظف مبينة فى كتلة منظمة تماماً ومعنونة بطريقة مرتبة والبيانات المدخلة موضوع تحتها خط .

```
                 MONTHLY PAYROLL

NAME ?ANDREWS, J J
EMPLOYEE NUMBER ?2717
GROSS SALARY ?870.00
MARITAL STATUS (M OR S) ?M
NUMBER OF EXEMPTIONS ?2
FEDERAL TAX=$ 108.984        STATE TAX=$ 10.05          LOCAL TAX=$ 8
NET SALARY=$ 742.966

NAME ?COHEN, A M
EMPLOYEE NUMBER ?5375
GROSS SALARY ?1250.00
MARITAL STATUS (M OR S) ?M
NUMBER OF EXEMPTIONS ?3
FEDERAL TAX=$ 170.886        STATE TAX=$ 15.75          LOCAL TAX=$ 8
NET SALARY=$ 1055.36

NAME ?DIPASQUALE, G V
EMPLOYEE NUMBER ?4660
GROSS SALARY ?2075.00
MARITAL STATUS (M OR S) ?S
NUMBER OF EXEMPTIONS ?1
FEDERAL TAX=$ 425.832        STATE TAX=$ 28.5           LOCAL TAX=$ 8
NET SALARY=$ 1612.67

NAME ?HOLLAND, C J
EMPLOYEE NUMBER ?0892
GROSS SALARY ?520.00
MARITAL STATUS (M OR S) ?S
NUMBER OF EXEMPTIONS ?2
FEDERAL TAX=$ 59.5           STATE TAX=$ 5.2            LOCAL TAX=$
 5.2
NET SALARY=$ 450.1

NAME ?JONES, D M
EMPLOYEE NUMBER ?6839
GROSS SALARY ?1120.00
MARITAL STATUS (M OR S) ?M
NUMBER OF EXEMPTIONS ?2
FEDERAL TAX=$ 156.484        STATE TAX=$ 13.8           LOCAL TAX=$ 8
NET SALARY=$ 941.716

NAME ?KOWALSKI, S
EMPLOYEE NUMBER ?8462
GROSS SALARY ?1100.00
MARITAL STATUS (M OR S) ?M
NUMBER OF EXEMPTIONS ?3
FEDERAL TAX=$ 142.386        STATE TAX=$ 13.5           LOCAL TAX=$ 8
NET SALARY=$ 936.114

NAME ?LOWE, H G
EMPLOYEE NUMBER ?9587
GROSS SALARY ?1075.00
MARITAL STATUS (M OR S) ?M
NUMBER OF EXEMPTIONS ?5
FEDERAL TAX=$ 117.04         STATE TAX=$ 13.125         LOCAL TAX=$ 8
NET SALARY=$ 936.835

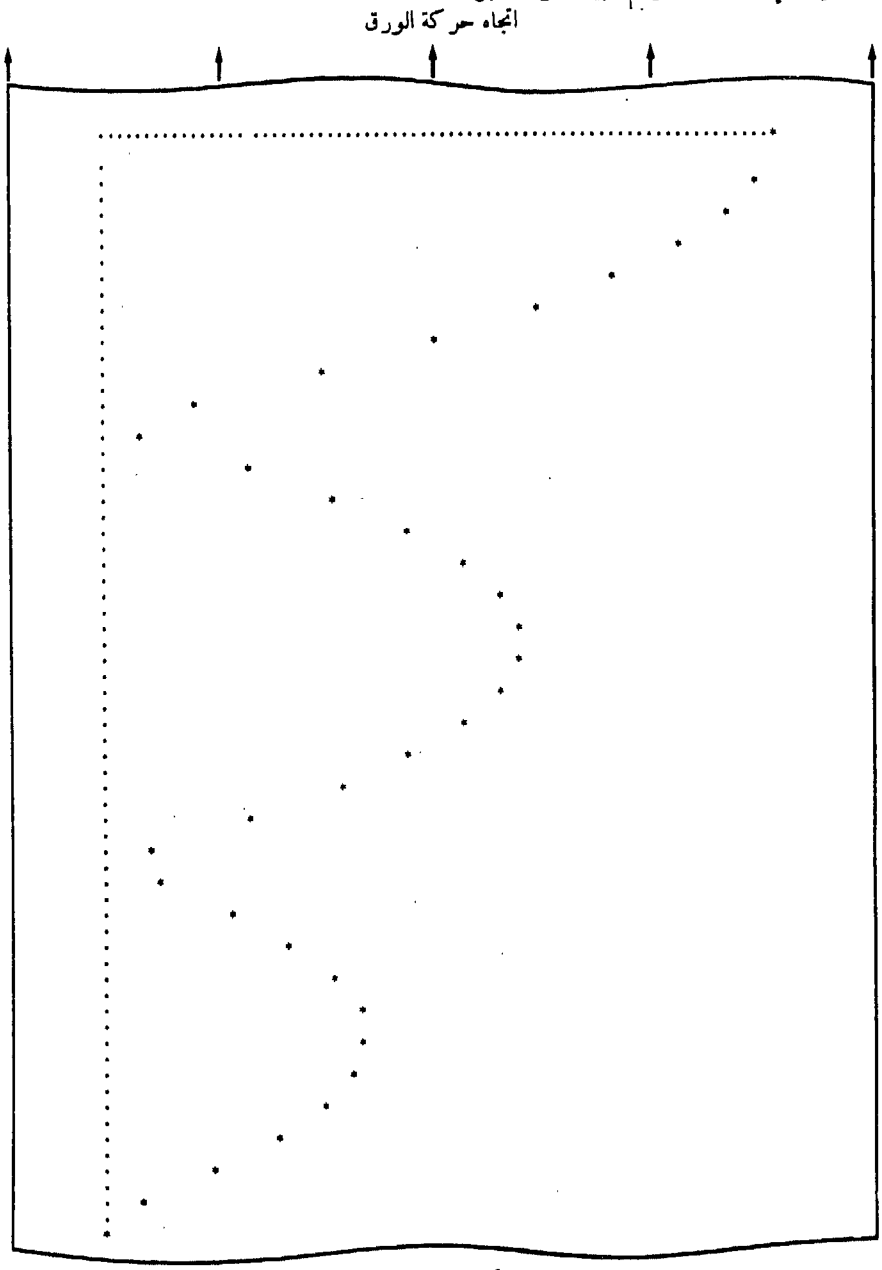
NAME ?END
```

شكل ٦ – ١٧

٦ ـــ ١٠ المخرجات البيانية GRAPHICAL OUTPUT

تولد عدة برامج لحاسب قائمة رقية من البيانات المخرجة والتى ترسم بيانياً فى آخر الأمر على ورق بيانى . وفى مثل هذه الحالات يكون غالباً أكثر فاعلية أن يكون الحاسب هو المسئول عن توليد المخرج البيانى مباشرة على ورق مثبت على النهاية الطرفية ، وذلك بالإضافة إلى المخرجات الرقية العادية . ويسمح ذلك للمستفيد بأن يرى المظهر العام للرسم البيانى ثم يرجع إلى الجداول الأكثر دقة للبيانات إذا تطلب الأمر ذلك .

إنه لغاية فى السهولة إصدار رسم بيانى على ورق مثبت على النهاية الطرفية ولتنفيذ ذلك يجب أن نستفيد من استخدام حلقة FOR-TO التكرارية ، والتى تتضمن جملة PRINT محتوية على دالة TAB . الرسم البيانى المنتج بهذه الطريقة سوف ينفذ رأسياً متجهاً إلى أسفل الصفحة المطبوعة وفيها المنحنى الحقيق ممثل بعدة نقط متصلة ومتساوية التباعد وقريبة من بعضها الآخر ، كما هو مبين فى شكل ٦ -١٨ . الميكانيكية التفصيلية لإصدار هذا الرسم البيانى موصفة بواسطة مثال .



شكل ٦ - ١٨

مثال ٦ – ٢٧

لقد تم توليد قائمتين رقميتين Y و T بواسطة برنامج بيسك ، الذى يصف موضع قذيفة فى أوقات متعددة . عناصر Y تمثل ارتفاع القذيفة وعناصر T تمثل الأوقات المناظرة) ونرغب فى إصدار صورة بيانية لقيم Y مقابل قيم T ، وذلك على ورق مثبت على النهاية الطرفية بعرض 72 حرفاً . ومن أجل توليد رسم بيانى واضح بقدر الإمكان سوف نضاعف المسافات بين النقط على المنحنى ، أى ، سوف نترك سطراً خالياً بين كل نجمتين (*) كما هو مبين فى شكل ٦ – ١٨ .

وبما أن الرسم البيانى سوف يكون رأسياً بطول الصفحة ، فإن محور Y سوف يولد بواسطة سطر واحد من النقط ومحور T سوف يكون عمودياً على السطر المطبوع . (أى أسفل الصفحة) . من ثم فسوف تطبع نقطة واحدة من محور T بواسطة كل سطر من المخرجات .

دعنا نطبع محور Y مبيناً موضع القذيفة عند بداية المحور (عند الزمن صفر) . ولعمل ذلك فإننا نطبع نقطة فى كل من الـ 71 موضع الأولى ثم بعد ذلك نضع النجمة فى الموضع رقم 72 وبذلك يمكن كتابة :

```
580 FOR J=0 TO 70
590       PRINT TAB(J);".";
600 NEXT J
610 PRINT TAB(71);"*"
620 PRINT "."
```

سوف تنتج الجمل من 580 إلى 610 محور Y ، كما هو موصوف أعلاه . وسوف يولد سطر متعاقب يحتوى على نقطة واحدة فقط فى أول عمود (تمثل جزءاً من محور T) وذلك بواسطة الجملة 620 . وهذا السطر مطلوب من أجل الحصول على التباعد المزدوج المطلوب .

بعد ذلك نرغب فى طباعة موضع القذيفة فى أوقات مختلفة . دعنا نولد سطرين للمخرجات فى كل مرة . أولهما يحتوى على نقطة (تمثل محور T) فى أول عمود ونجمة (تمثل موضع القذيفة) فى وضع ملائم على طول السطر . ويترك السطر التالى خالياً ويحتوى فقط على نقطة فى أول موضع . ولإنجاز ذلك يمكننا كتابة :

```
630 FOR I=2 TO I1
640       LET J=INT(71*Y(I)/Y(1))
650       IF J=0 THEN 680
660       PRINT ".";TAB(J);"*"
670       GO TO 690
680       PRINT "*"
690       PRINT "."
700 NEXT I
```

حيث I1 رقم صحيح يبين آخر نقطة مطلوب رسمها (آخر وقت) . لاحظ أن أول سطر فى الطباعة سوف ينفذ بالجمل من 640 إلى 680 ، والسطر الثانى يولد بواسطة الجملة 690 .

الجملة :

```
640       LET J=INT(71*Y(I)/Y(1))
```

يمكن أن تتطلب توضيحاً إضافياً أولا ، لاحظ أن Y(1) هى الارتفاع الحقيق للقذيفة . إذا فرضنا أن ذلك يمثل أقصى ارتفاع للقذيفة ، فإن Y(I) وهو ارتفاع القذيفة عند الوقت رقم I سوف يكون رقماً ما محصوراً بين صفر و Y(1) ونبحث عن قيمة للمتغير J رقم صحيح ينحصر مابين 71,0 وسوف تشير قيمة J إلى مكان طباعة النجمة على الورقة ، أى J = 0 تناظر Y(I) = 0 و J = 71 تناظر Y(I) = Y(1) من ثم ينسب مباشرة :

$$\frac{J}{71} = \frac{Y(I)}{Y(1)} \qquad \text{or} \qquad J=71*Y(I)/Y(1)$$

حيث يمكن أن تأخذ J أرقاماً صحيحة فقط ، من ثم نكتب

```
J=INT(71*Y(I)/Y(1))
```

فى المثال ٦ – ٢٨ سوف نرى برنامج بيسك كاملا يولد كلا من النتائج الرقية والرسم البيانى .

مثال ٦ – ٢٨ محاكاة ارتداد كرة Simulation of a Bouncing Ball

فى هذا المثال ، نود حساب حركة كرة من المطاط عند ارتدادها من أعلى إلى أسفل تحت تأثير الجاذبية الأرضية ، بينما فى نفس الوقت تسير فى الاتجاه الأفقى بسرعة منتظمة . سوف نفرض أن الزحزحة الرأسية المبدئية (H) (أى ، الارتفاع الأساسى أعلىالأرض) والسرعة الأفقية (V) وعدد مرات ارتداد الكرة (N) . وأيضاً سوف يكون معامل الارتداد (C) معلوماً وهو النسبة بين السرعة الرأسية بعد الصدمة مباشرة إلى السرعة الرأسية قبل الصدمة مباشرة .

طريقة إجـراء الحسابات Computational Procedure

من أجل حساب موضع الكرة عند أوقات متعددة ، سوف نختار زيادة صغيرة فى الوقت (D) ، ثم بعد ذلك نستخدم قوانين الطبيعة التالية التى تطبق مع كل زيادة فى الوقت :

```
T(I+1)=T(I)+D
X(I+1)=X(I)+V*D
Z(I+1)=Z(I)-G*D
Y(I+1)=Y(I)+.5*(Z(I)+Z(I+1))*D
```

حيث تشير X إلى الزحزحة الأفقية (تبدأ بصفر عند بداية البرنامج) ، وتشير Z إلى السرعة الرأسية (وهى أيضاً صفر عند بدايةالبرنامج) و Y هى الارتفاع فوق سطح الأرض و G هى العجلة الناتجة عن الجاذبية الأرضية (32.2 قدم / ثانية ٢) . تشير الأدلة I و I + 1 لقيم المتغيرات المختلفة عند بداية ونهاية الزيادة فى الزمن على الترتيب .

إذا حدث أى ارتداد أثناء الزيادة فى الزمن . فن الضرورى تعديل الصيغ الحسابية فشرط الارتداد يميز بقيمة سالبة للمتغير Y(I + 1) والذى يستحيل فى التطبيقات الطبيعية . عند حدوث هذا الشرط فإننا نعيد حساب Z(I + 1) و Y(I + 1) كما يلى . أولا احسب الزمن اللازم لاصطدام الكرة بالأرض ، مبتدئاً من مكانها عند بداية زيادة الزمن للمرة I إذا أطلقنا على هذا الزمن D1 فإنه من التناسب البسيط :

$$\frac{D1}{D} = \frac{Y(I)-0}{Y(I)-Y(I+1)}$$

حيث يمكن كتابته فى البيسك كما يلى

```
D1=D*Y(I)/(Y(I)-Y(I+1))
```

يمكننا حساب السرعة الرأسية فوراً قبل الاصطدام .

```
Z=Z(I)-G*D1
```

وبذلك فسوف تكون السرعة الرأسية بعد نهاية زيادة الزمن .

```
Z1=-C*(Z(I)-G*D1)
```

وإلا سوف تكون السرعة الرأسية عند نهاية زيادة الزمن .

```
Z(I+1)=Z1-G*(D-D1)
```

والزحزحة الرأسية عند نهاية زيادة الوقت يمكن كتابتها كما يلى :

```
Y(I+1)=.5*(Z1+Z(I+1))*(D-D1)
```

المخطط التمهيدى للبرنامج The Program Outline

والآن لدينا معلومات كافية لكتابة مخطط تمهيدى لبرنامج بيسك كامل بالتفصيل :

١ – إقرأ H و V و N و C و D

(أ) إذا كانت $H = 0$ إذن أنه البرنامج

(ب) إذا كانت H لها أى قيمة موجبة فواصل للنقط التالية .

٢ – إبدأ بإعطاء قيم أولية لكل المعاملات :

$I = 1$ (I هو عداد الزيادة)	$X(1) = 0$
$B = 0$ (B هو عداد الارتداد)	$Z(1) = 0$
$T(1) = 0$	$Y(1) = H$

٣ – احسب الزحزحة الرأسية والأفقية والسرعة الرأسية لكل زيادة فى الزمن باستخدام الصيغ الرياضية التى سبق ذكرها .

٤ – إذا اصطدمت الكرة بالأرض أثناء الزيادة فى الزمن ، اختبر لترى هل هذا شرط ارتداد أو إنهاء برنامج .

(أ) شرط ارتداد $(B < N)$ – وبذلك أعد حساب السرعة الرأسية والزحزحة الرأسية لتأخذ فى الاعتبار الارتداد ، ثم زد عداد الارتداد (أى $B = B + 1$) ثم بعد ذلك أكمل إلى الزيادة التالية فى الزمن .

(ب) شرط الانتهاء $(B = N)$ – أوجد الزمن النهائى والزحزحة الأفقية عندما تصطدم الكرة بالأرض .

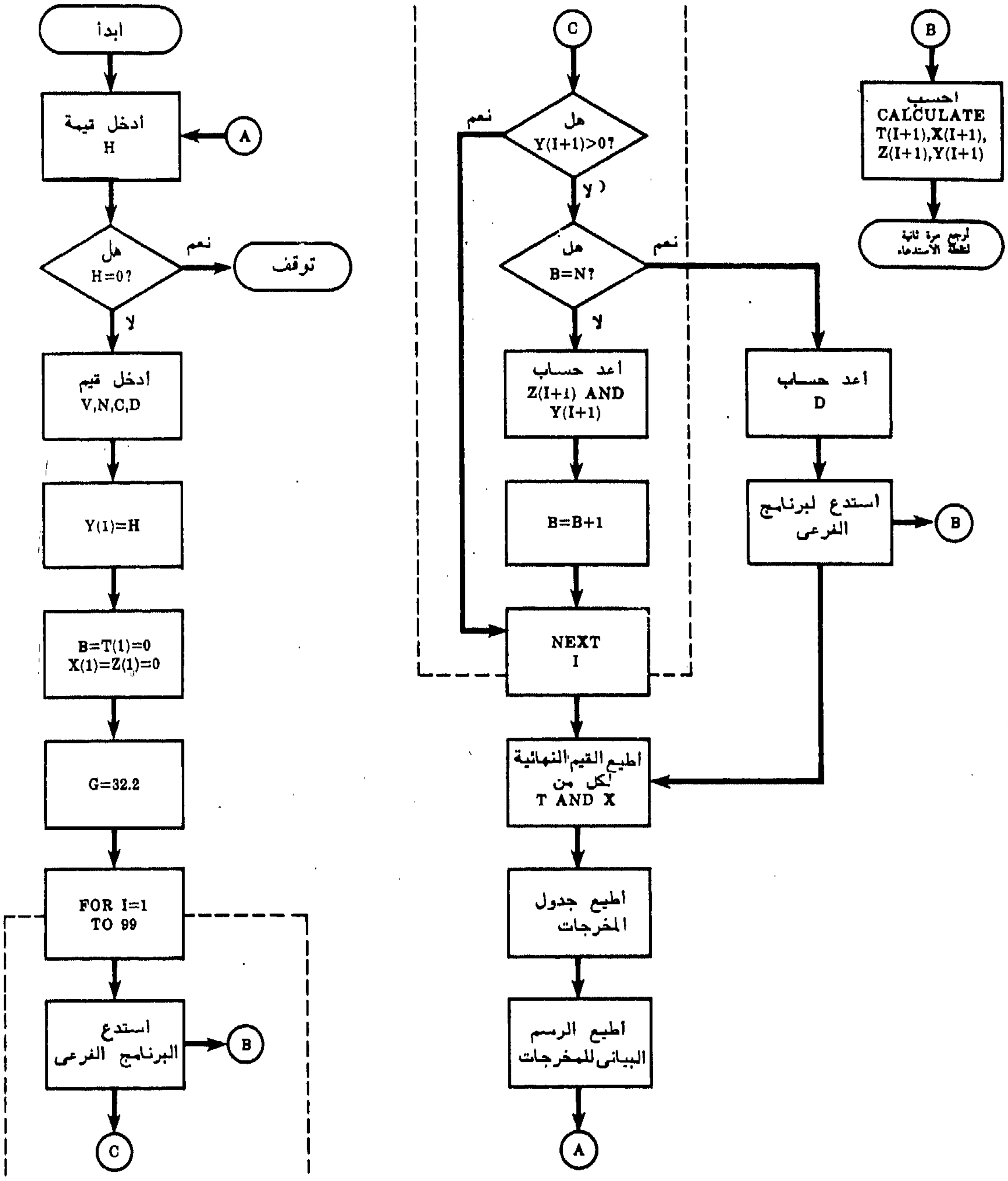
٥ – اطبع القيم النهائية لكل من X و T يليها جدولة كاملة لكل من T و X و Y و Z .

٦ – ارسم Y مقابل T باستخدام الطريقة التى تمت مناقشتها فى المثال ٦ – ٢٧ .

٧ – ارجع مرة ثانية للخطوة ١ .

يبين شكل ٦ – ١٩ . خريطة سير عمليات كاملة لهذه الحسابات . لاحظ أن الحسابات لكل من $T(I + 1)$ و$X(I + 1)$ و $Z(I + 1)$ و $Y(I + 1)$ مبينة بداخل هيكل البرنامج الفرعى .

يجب أن يكتشف القارىء الذى درس التكامل بالطرق العددية أننا فى هذا المثال نكامل المعادلة التفاضلية من الدرجة الثانية $d^2y/dt^2 = -g$ وتعرف طريقة التكامل باسم **طريقة أويلر المعدلة** ***modified Euler method***


ابدأ
أدخل قيمة H
A
هل H=0?
نعم
توقف
لا
أدخل قيم V,N,C,D
Y(1)=H
B=T(1)=0
X(1)=Z(1)=0
G=32.2
FOR I=1 TO 99
استدع البرنامج الفرعى
B
C
C
هل Y(I+1)>0?
نعم
لا
هل B=N?
نعم
لا
أعد حساب Z(I+1) AND Y(I+1)
B=B+1
NEXT I
أعد حساب D
استدع لبرنامج الفرعى
B
اطبع القيم النهائية لكل من T AND X
اطبع جدول المخرجات
اطبع الرسم البيانى للمخرجات
A
B
احسب
CALCULATE
T(I+1),X(I+1),
Z(I+1),Y(I+1)
ارجع مرة ثانية لنقطة الاستدعاء

شكل ٦ – ١٩

```
10 REM          SIMULATION OF A BOUNCING BALL
20 DIM X(100),Y(100),Z(100),T(100)
30 PRINT "INITIAL HEIGHT OF BALL (FT)";
40 INPUT H
50 IF H=0 THEN 810
60 PRINT "INITIAL HORIZONTAL VELOCITY (FT/SEC)";
70 INPUT V
80 PRINT "NUMBER OF BOUNCES";
90 INPUT N
100 PRINT "BOUNCE COEFFICIENT";
110 INPUT C
120 PRINT "LENGTH OF TIME INCREMENT (SEC)";
130 INPUT D
140 PRINT
150
160 REM          INITIALIZE PARAMETERS
170
180 LET B=T(1)=X(1)=Z(1)=0
190 LET Y(1)=H
200 LET G=32.2
210
220 REM COMPUTE VELOCITY AND DISPLACEMENT FOR EACH TIME INCREMENT
230
240 FOR I=1 TO 99
250    GOSUB 730
260    IF Y(I+1)>0 THEN 330
270    IF B=N THEN 360
280    LET D1=D*Y(I)/(Y(I)-Y(I+1))        'CORRECT FOR BOUNCE CONDITION
290    LET Z1=-C*(Z(I)-G*D1)
300    LET Z(I+1)=Z1-G*(D-D1)
310    LET Y(I+1)=.5*(Z1+Z(I+1))*(D-D1)
320    LET B=B+1
330 NEXT I
340 GOTO 400
350
360 REM          BALL HITS GROUND FOR LAST TIME
370
380 LET D=D*Y(I)/(Y(I)-Y(I+1))
390 GOSUB 730
400 LET I1=I+1
410 LET T1=T(I1)
420 LET X1=X(I1)
430
440 REM          PRINT NUMERICAL OUTPUT
450
460 PRINT "HORIZONTAL DISTANCE TRAVELED=";X1;"FT"
470 PRINT "TIME REQUIRED=";T1;"SECS"
480 PRINT
490 FOR I=1 TO I1
500    PRINT "T=";T(I),"X=";X(I),"Y=";Y(I),"Z=";Z(I)
510 NEXT I
520 PRINT
530
540 REM          PRINT GRAPHICAL OUTPUT
550
560 PRINT "GRAPHICAL SOLUTION TO BOUNCING BALL PROBLEM"
570 PRINT
580 FOR J=0 TO 70
590    PRINT TAB(J);".";
600 NEXT J
610 PRINT TAB(71);"*"
620 PRINT "."
630 FOR I=2 TO I1        'GENERATE SUCCESSIVE POINTS OF CURVE
640    LET J=INT(71*Y(I)/Y(1))
650    IF J=0 THEN 680
660    PRINT ".";TAB(J);"*"
670    GOTO 690
680    PRINT "*"
690    PRINT "."
700 NEXT I
710 GOTO 30
720
730 REM SUBROUTINE TO CALCULATE VELOCITY AND DISPLACEMENT AT END
740 REM          OF TIME INCREMENT
750
760 LET T(I+1)=T(I)+D
770 LET X(I+1)=X(I)+V*D
780 LET Z(I+1)=Z(I)-G*D
790 LET Y(I+1)=Y(I)+.5*(Z(I+1)+Z(I))*D
800 RETURN
810 END
```

شكل ٦ — ٢٠

برنامج بيسك The BASIC Program

يبين شكل ٦ – ٢٠ برنامج بيسك كاملا للقيام بهذه العمليات الحسابية . يسمح البرنامج بعدد من الزيادات المتتالية للزمن تصل إلى 100 مرة ، وبذلك فإن طول فترة الزيادة الزمنية (D) يجب أن تختار كبيرة نسبياً حتى لاتتعدى عدد الزيادات الزمنية هذا الرقم . (ومن الناحية الأخرى . فإن D لا يمكن أن نعطيها قيمة كبيرة وألا أصبح من غير الممكن تطبيق الصيغة الرياضية التى تحسب $T(I + 1)$ و $X(I + 1)$ و $Y(I + 1)$ و $Z(I + 1)$. وكقاعدة تقريبية يمكن أن يمثل كل ارتداد بواسطة 8 إلى 20 نقطة) .

والبرنامج الفرعى الذى يستخدم فى حساب $T(I + 1)$ و $X(I + 1)$ و $Z(I + 1)$ و $Y(I + 1)$ يتكون من الجمل 730 إلى 800 ويجب أن يكون مفهوماً أن هيكل البرنامج الفرعى ليس جوهرياً للقيام بهذه الحسابات . وأن استخدام البرنامج الفرعى يكون مرغوباً فيه حيث أن هذه الكتلة من الجمل قد استدعت من مكانين مختلفين بداخل البرنامج (جملتى 250 و 390) .

توليد جدول البيانات المخرجة ينفذ بطريقة مباشرة بواسطة الحلقة التكرارية FOR - TO (الجمل 490 إلى 510) . بالرغم من أن الطريقة المستخدمة لتوليد الرسم البيانى (الجمل 560 إلى 700) أقل وضوحاً ، فهذا الجزء من البرنامج مطابق تماماً لما سبقت مناقشته فى المثال ٦ – ٢٧ . ومن ثم فإن المنطق المتبع يجب أن يكون واضحاً .

الشكل ٦ – ٢١ (أ) يحتوى على المخرجات العددية التى تم توليدها لبيانات الإدخال التالية : (والبيانات المدخلة موضوع تحتها خط) .

$$H = 2.00 \text{ ft}, \ V = 1.20 \text{ ft/sec}, \ N = 3, \ C = 0.80, \ D = 0.05 \text{ sec}$$

يتضح أن المسافة المطلوبة حتى تكمل الكرة ثلاثة ارتدادات كاملة هى 2.06 قدم والزمن المناظر لذلك هو 1.72 ثانية .

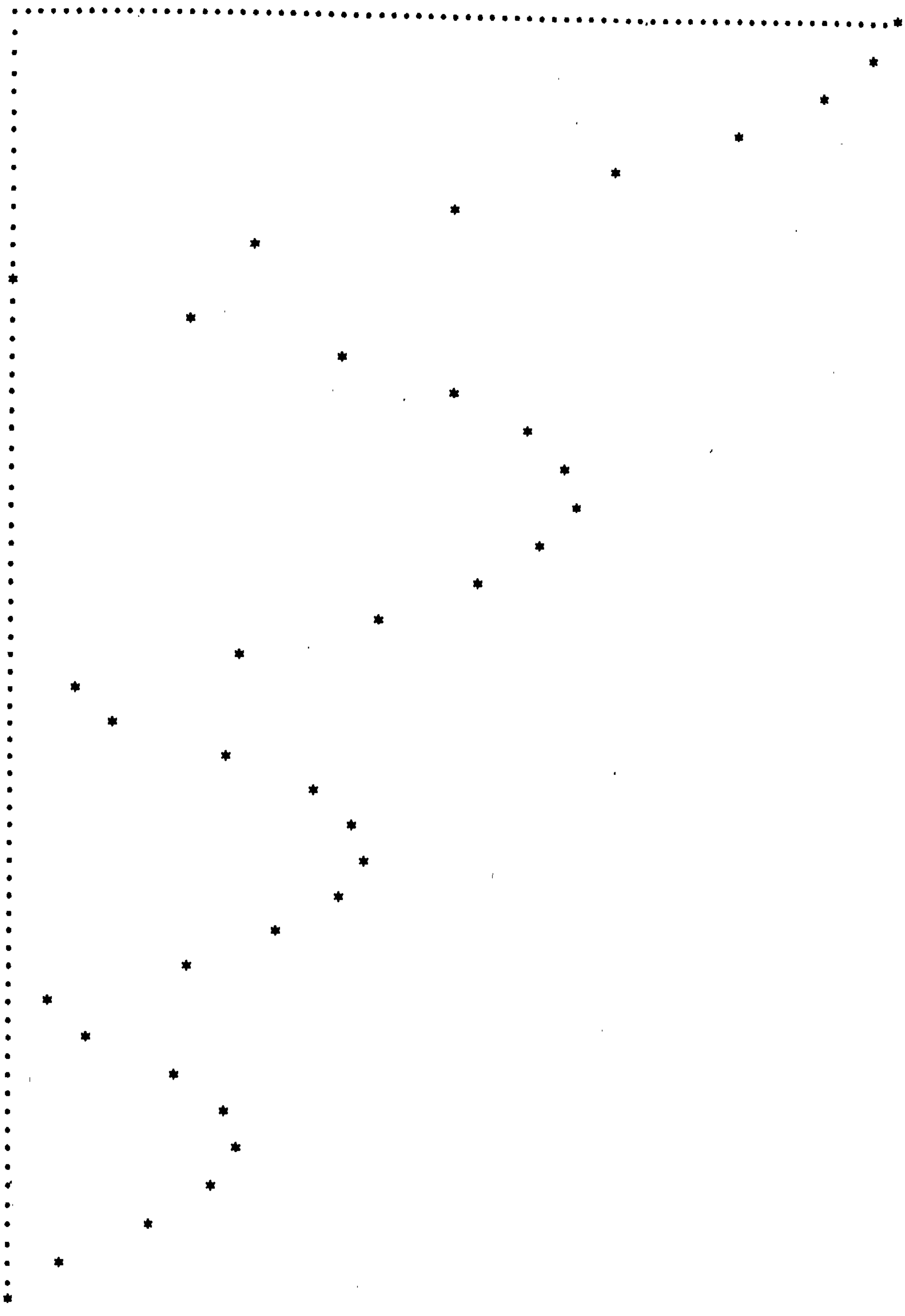
```
INITIAL HEIGHT OF BALL (FT) ?2.00
INITIAL HORIZONTAL VELOCITY (FT/SEC) ?1.20
NUMBER OF BOUNCES ?3
BOUNCE COEFFICIENT ?0.80
LENGTH OF TIME INCREMENT (SEC) ?0.05


HORIZONTAL DISTANCE TRAVELED= 2.06344 FT
TIME REQUIRED= 1.71953 SECS


T= 0          X= 0            Y= 2            Z= 0
T= 0.05       X= 6.00000E-2   Y= 1.95975      Z=-1.61
T= 0.1        X= 0.12         Y= 1.839        Z=-3.22
T= 0.15       X= 0.18         Y= 1.63775      Z=-4.83
T= 0.2        X= 0.24         Y= 1.356        Z=-6.44
T= 0.25       X= 0.3          Y= 0.99375      Z=-8.05
T= 0.3        X= 0.36         Y= 0.551        Z=-9.66
T= 0.35       X= 0.42         Y= 2.77500E-2   Z=-11.27
T= 0.4        X= 0.48         Y= 0.396269     Z= 7.5392
T= 0.45       X= 0.54         Y= 0.732979     Z= 5.9292
T= 0.5        X= 0.6          Y= 0.989189     Z= 4.3192
T= 0.55       X= 0.66         Y= 1.1649       Z= 2.7092
T= 0.6        X= 0.72         Y= 1.26011      Z= 1.0992
T= 0.65       X= 0.78         Y= 1.27482      Z=-0.5108
T= 0.7        X= 0.84         Y= 1.20903      Z=-2.1208
T= 0.75       X= 0.9          Y= 1.06274      Z=-3.7308
T= 0.8        X= 0.96         Y= 0.835949     Z=-5.3408
T= 0.85       X= 1.02         Y= 0.528659     Z=-6.9508
T= 0.9        X= 1.08         Y= 0.140869     Z=-8.5608
T= 0.95       X= 1.14         Y= 0.233292     Z= 6.1104
T= 1.         X= 1.2          Y= 0.498562     Z= 4.5004
T= 1.05       X= 1.26         Y= 0.683332     Z= 2.8904
T= 1.1        X= 1.32         Y= 0.787602     Z= 1.2804
T= 1.15       X= 1.38         Y= 0.811372     Z=-0.329597
T= 1.2        X= 1.44         Y= 0.754642     Z=-1.9396
T= 1.25       X= 1.5          Y= 0.617413     Z=-3.5496
T= 1.3        X= 1.56         Y= 0.399683     Z=-5.1596
T= 1.35       X= 1.62         Y= 0.101453     Z=-6.7696
T= 1.4        X= 1.68         Y= 0.189303     Z= 4.58198
T= 1.45       X= 1.74         Y= 0.378152     Z= 2.97198
T= 1.5        X= 1.8          Y= 0.486501     Z= 1.36198
T= 1.55       X= 1.86         Y= 0.514351     Z=-0.248016
T= 1.6        X= 1.92         Y= 0.4617       Z=-1.85802
T= 1.65       X= 1.98         Y= 0.328549     Z=-3.46802
T= 1.7        X= 2.04         Y= 0.114898     Z=-5.07802
T= 1.71953    X= 2.06344      Y= 9.58084E-3   Z=-5.7069
```

شكل ٦ – ٢١ (أ)

GRAPHICAL SOLUTION TO BOUNCING BALL PROBLEM



INITIAL HEIGHT OF BALL (FT) ?0

شكل ٦ – ٢١ (ب)

نرى فى شكل ٦ – ٢١ (ب) رسماً بيانياً مجهزاً بالحاسب موضحاً أماكن الكرة فى أزمنة متعددة . ويمكن أن ترى الارتدادات الفردية بكل وضوح فى هذا الشكل .

وأخيراً نرى فى نهاية شكل ٦ – ٢١ (ب) طلبا لمجموعة جديدة من بيانات الإدخال . وتتوقف الحسابات إذا أعطينا قيمة صفر للمتغير H وسوف نتعرض لتوليد المخرجات البيانية بشكل أوسع فى الفصل الثانى عشر الذى يختص ببيانات الحاسب الدقيق .

اسئلة للمراجعة
Review Questions

٦ – ١ ماهى أوجه الاختلاف بين الدالة والبرنامج الفرعى ؟

٦ – ٢ هل الدوال والبرامج الفرعية دائماً مطلوبة فى برامج بيسك ، ماهى المزايا فى استخدامها ؟

٦ – ٣ ماهى الطريقة التى يمكن أن تسهم بها الدوال والبرامج الفرعية فى تطوير تنظيم برنامج بيسك .

٦ – ٤ لخص قواعد تسمية الدوال . وكيف يمكن تمييز الدوال الرقمية عن الدوال الحرفية ؟

٦ – ٥ ماهو الغرض من جملة DEF ؟ وكيف تكتب ؟

٦ – ٦ ماهى القواعد التى تحكم استخدام الخلاصات فى الدوال ؟ وهل يمكن للدالة أن تستخدم متغيرات غير موصوفة كخلاصات بها ؟

٦ – ٧ ماهو الفرق بين تعريف الدالة والإشارة إليها ؟ وكيف يمكن الإشارة لدالة ؟

٦ – ٨ ماهى القيم الزائفة ؟ وما هو التناظر الذى يجب وجوده بين مجموعة من الخلاصات عند الإشارة للدالة والخلاصات الزائفة المزاملة لها ؟

٦ – ٩ هل يمكن للخلاصة أن تتكون من أى شىء خلاف متغير ليس له دليل (على سبيل المثال : الثابت ، المتغير ذو الدليل أو الصيغة الرياضية) ؟ هل هذا صحيح أيضاً لخلاصة زائفة ؟

٦ – ١٠ كيف تكتب جملة DEF فى دالة متعددة الأسطر ؟

٦ – ١١ اذكر مكانين يجب أن يظهر فيهما اسم الدالة المتعددة الأسطر ؟

٦ – ١٢ ماهو الغرض من جملة FNEND ؟ وكيف تكتب ؟

٦ – ١٣ هل يمكن تحويل التحكم خارج الدالة متعددة الأسطر بواسطة جملة GO TO أو جملة RETURN ؟

٦ – ١٤ هل يجب أن تكون القيمة الناتجة من الدالة من نفس نوع خلاصات الدالة ؟

٦ – ١٥ هل يجب أن تكون كل خلاصات الدلة من نفس النوع ؟

٦ – ١٦ كيف تخزن الحروف بداخل الحاسب ؟

٦ – ١٧ ما هو كود ASCII ذو الأرقام السبعة الثنائية ؟

٦ – ١٨ ماهو الغرض من جملة CHANGE ؟ اذكر طريقتين يمكن أن تكتب بهما ؟

٦ — ١٩ ماهو الغرض من دالة ASC ؟ و كيف تستخدم ؟

٦ — ٢٠ ماهو الغرض من دالة CHR $ ؟ و كيف تستخدم ؟

٦ — ٢١ ماهو الغرض من دالة RND ؟ و كيف تستخدم ؟ وهل تتطلب هذه الدالة خلاصة ؟

٦ — ٢٢ ماهو المقصود بالأرقام العشوائية الزائفة ؟ وكيف تختلف الأرقام العشوائية الزائفة عن الأرقام التى هى فى حقيقة الأمر عشوائية ؟

٦ — ٢٣ ماهو الغرض من جملة RANDOMIZE ؟ و كيف تكتب ؟

٦ — ٢٤ لخص القواعد الخاصة بتعريف البرامج الفرعية . هل يجب أن يبدأ البرنامج الفرعى بجملة معينة ؟

٦ — ٢٥ هل يمكن تضمين خلاصات فى البرنامج الفرعى ؟

٦ — ٢٦ هل يمكن أن ينتهى برنامج فرعى بجملة FNEND ؟ أو جملة RETURN ؟ أو جملة END ؟

٦ — ٢٧ ماهو الغرض من جملة RETURN ؟ وكيف تكتب ؟ ماذا يحدث عند مقابلة جملة RETURN أثناء تنفيذ البرنامج ؟

٦ — ٢٨ هل يمكن أن يحتوى البرنامج الفرعى على أكثر من جملة RETURN ؟ إشرح .

٦ — ٢٩ ماهو الغرض من جملة GOSUB ؟ و كيف تكتب ؟ هل يمكن لبرنامج يحتوى على برنامج فرعى واحد أن يحتوى على أكثر من جملة GOSUB ؟

٦ — ٣٠ هل يمكن تضمين حلقة تكرارية FOR—TO فى برنامج فرعى أو دالة متعددة الأسطر ؟

٦ — ٣١ هل يمكن تحويل التحكم إلى خارج البرنامج الفرعى بواسطة جملة GO TO ؟ أو جملة IF-THEN ؟

٦ — ٣٢ صف الترتيب الهرمى الذى يجب أن يلاحظ عند تداخل البرامج الفرعية ؟

٦ — ٣٣ ماهى مزايا عرض البيانات المخرجة بيانياً ؟

٦ — ٣٤ ماهى الدوال المكتبية التى تستخدم لإنتاج مخرجات بيانية ؟ وبأى نوع من البرمجة الهيكلية يمكن تضمين هذه الدوال عادة ؟

مسائل محلولة
Solved Problems

٦ — ٣٥ اكتب دالة بيسك لكل من المواقف التالية :

(أ) احسب قيمة الصيغة الجبرية : $z = \dfrac{(u/v) + (x/y)}{2}$

```
10 DEF FNZ(U,V,X,Y)=(U/V+X/Y)/2
```

(ب) إذا كانت X ممثلة بكمية كسرية موجبة ، أوجد قيمة مقربة للقيمة X لها رقمان عشريان يمين العلامة العشرية .

```
20 DEF FNY(X)=.01*INT(100*(X+.005))
```

(ج) احسب قيمة الصيغة الجبرية :

$$p = \begin{cases} \log(t^2 - a) & \text{for } t^2 > a \\ \log(t^2) & \text{for } t^2 \leq a \end{cases}$$

```
30 DEF FNP(T,A)
40 IF T<=A THEN 70
50 LET FNP=LOG(T↑2-A)
60 GO TO 80
70 LET FNP=LOG(T↑2)
80 FNEND
```

(د) احسب مجموع الـ N عنصراً الأولى فى قائمة رقية L ، أى احسب مجموع L(1) + L(2) + L(3) + ... + L(N)

```
100 DEF FNS(N)
110 LET S=0
120 FOR I=1 TO N
130     LET S=S+L(I)
140 NEXT I
150 LET FNS=S
160 FNEND
```

(هـ) نفرض أن كلا من M$ و N$ تمثل حرفاً واحداً . أنشىء سلسلة حرفية واحدة تحتوى هذين الحرفين مرتبة ترتيباً أبجدياً .

```
200 DEF FNN$(M$,N$)
210 LET L(0)=2
220 IF M$>N$ THEN 260
230 LET L(1)=ASC(M$)
240 LET L(2)=ASC(N$)
250 GO TO 280
260 LET L(1)=ASC(N$)
270 LET L(2)=ASC(M$)
280 CHANGE L TO L$
290 LET FNN$=L$
300 FNEND
```

(و) احسب متوسط رقين عشوائيين كل منهما له قيمة تتراوح مابين A و B .

```
100 DEF FNR(A,B)
110 LET R1=A+(B-A)*RND
120 LET R2=A+(B-A)*RND
130 LET FNR=(R1+R2)/2
140 FNEND
```

يمكن أن تكتب هذه الدالة أيضاً

```
100 DEF FNR=A+(B-A)*(RND+RND)/2
```

٦ - ٣٦ كل من المواقف الموصوفة التالية تتطلب استدعاء إحدى الدوال المعرفة فى المسألة ٦ - ٣٥ . اكتب جملة بيسك المناسبة أو مجموعة من الجمل المتتالية ، فى كل حالة .

(أ) اطبع قيمة f ، حيث

$$f = \frac{(a/b) + (c/d)}{2}$$

(أنظر المسألة ٦ – ٣٥ (أ))

```
100 PRINT FNZ(A,B,C,D)
```

(ب) نفرض أن T تمثل كية موجبة يمكن أن تزيد قيمتها عن الواحد احسب قيمة T1 ، حيث T1 لها نفس قيمة T ماعداً أن الجزء الكسرى فى T1 يقرب إلى أقرب رقمين عشرين .

(أنظر المسألة ٦ – ٣٥ (ب)) .

```
110 LET T1=INT(T)+FNY(T-INT(T))
```

(حـ) دع P1 تمثل الكية

$$\log[(a+b)^2 - c] \quad \text{if} \quad (a+b)^2 > c$$

و

$$\log[(a+b)^2] \quad \text{if} \quad (a+b)^2 \leq c$$

(أنظر المسألة ٦ – ٣٥ (حـ) .)

```
30 LET P1=FNP((A+B),C)
```

(د) قائمة عددية L بها 101 عنصرا .مبتدئا بالعنصر (١) L ، حدد كم عدد العناصر المتعاقبة التى يمكن أضافتها بدون أن يتعدى المجموع القيمة 25 (انظر مسألة ٦ – ٣٥ (د)) .

```
40 FOR J=1 TO 100
50     IF FNS(J)>25 THEN 80
60 NEXT J
70 LET J=101
80 PRINT "N=";J-1
```

(هـ) كل من المتغيرات M و N تمثل مكافئ كود ASCII لحرف واحد . كون سلسلة حرفية تحتوى على حرفين بترتيب ابجدى (انظر المسألة ٦ – ٣٥ (هـ)) .

```
80 LET L$=FNN$(CHR$(M),CHR$(N))
```

(و) حدد متوسط رقمين عشوائيين . كل منهما له قيمة ما بين 1 و 10 . حول التحكم إلى الجملة رقم 250 إذا كان المتوسط يتعدى 5 . (انظر المسألة ٦ – ٣٥ (و)) .

```
100 IF FNR(1,10)>5 THEN 250
```

٦ – ٣٧ تحتوى كل من المسائل التالية على تعريف دالة و / أو استدعاء دالة حيث كتبت بصورة غير صحيحة . تعرف على كل الأخطاء .

(أ)

```
10 DEF FNW(A,B,C↑2,3)=((A+B)*C↑2)/3
```

لا يمكن استخدام الثوابت والصيغ الرياضية كخلاصات زائفة .

(ب)

```
10 DEF FNC(T1,T2,N)=((T1-T2)/T2)↑N
   . . .
60 LET V=C*FNC(2*A,F$)
```

لا تتوافق الخلاصات فى الإشارة إلى الدالة مع الخلاصات الزائفة فى تعريف الدالة من حيث العدد والنوع .

```
10 DEF FN4(X(1),X(2),X(3))=X(1)+2*X(2)-3*X(3)
```

تحتوى هذه الجملة على خطأين :

(i) اسم الدالة غير صحيح .

(ii) لا يمكن أن تظهر المتغيرات ذات الأدلة كخلاصات زائفة .

```
10 DEF FNG(A,B,C)
20 LET P=A+B*X+C*X↑2
30 LET Q=B+C*X
40 LET G=P+Q*X+C*X↑2
50 FNEND
```

اسم الدالة FNG لم تحدد أى قيمة بداخل الدالة .

```
100 DEF FNC(X,Y,Z)
110 IF X+Y>Z THEN 140
120 LET FNC=LOG(Z-(X+Y))
130 RETURN
140 LET FNC=LOG(Z)
150 FNEND
```

لا يمكن تحويل التحكم خارج الدالة بواسطة جملة RETURN

٦ – ٣٨ مجموعة الجمل التالية تمثل اجزاء من برامج بيسك تحتوى على برنامج فرعى أو أكثر . وقد كتب مثال منها بصورة صحيحة .

(أ)

```
10 DIM L(100)
   . . .
60 GOSUB 200
   . . .
120 GOSUB 200
   . . .
160 GOSUB 200
   . . .
190 STOP
200 LET S=0                 ⎫
210 FOR I=1 TO N            ⎪
220       LET S=S+L(I)      ⎬  برنامج فرعى
230 NEXT I                  ⎪
240 RETURN                  ⎭
250 END
```

لاحظ أن البرنامج الفرعى قد تم استدعاؤه من ثلاث نقط مختلفة بداخل برنامج الحاسب .

(ب)

```
50 GOSUB 120
   ...
120 IF A>B THEN 150
130 LET C=SQR((B-A)↑N)
140 RETURN
150 LET C=SQR(((A+B)/(A-B))↑N)
160 RETURN
```

برنامج فرعى

يحتوى البرنامج الفرعى فى هذا المثال على جملتى RETURN

(جـ)

```
10 DEF FNZ(X,Y)=X↑2+Y↑2
   ...
70 GOSUB 180
   ...
180 REM SAMPLE SUBROUTINE
   ...
210 LET W=FNZ(A,B+C)
   ...
250 RETURN
```

برنامج فرعى

لاحظ أن البرنامج الفرعى يستدعى الدالة FNZ المعرفة بواسطة المبرمج .

(د)

```
75 GOSUB 300
   ...
125 GOSUB 200
   ...
200 LET Z=C1*X+C2*Y
   ...
250 GOSUB 300
   ...
290 RETURN
300 LET W=(U+V)/Z
   ...
370 RETURN
380 END
```

البرنامج الفرعى الأول (200–290)

البرنامج الفرعى الثانى (300–370)

يستخدم هذا المثال البرامج المساعدة المتداخلة.لاحظ أن البرنامج الفرعى الثانى يستدعى من البرنامج الفرعى الأول ومن الجزء الأساسى من البرنامج .

٦ – ٣٩ تمثل المجموعات التالية من الجمل أجزاء من برامج بيسك والتى تحتوى على برامج فرعية . كل مثال يحتوى على خطأ أو أكثر . تعرف على كل الأخطاء .

(أ)

```
45 GOSUB 165
   ...
165 LET C=C1+C2+C3
   ...
190 GO TO 60
   ...
225 RETURN
230 FNEND
235 END
```

البرنامج الفرعى

يحتوى هذا البرنامج على خطأين :

(i) لا يمكن تحويل التحكم خارج البرنامج الفرعى بواسطة جملة GO TO .

(ii) لا يمكن أن ننهى البرنامج الفرعى بجملة FNEND .

(ب)

```
60 GOSUB 200
   ...
120 IF X<Y THEN 225
   ...
200 REM START OF SUBROUTINE  }
   ...                        } البرنامج الفرعى
300 RETURN                   }
```

لا يمكن تحويل التحكم داخل البرنامج الفرعى بواسطة جملةIF-THEN .

(جـ)

```
30 GOSUB 100
   ...
100 REM SUBROUTINE A  }
   ...                 }
120 GOSUB 200          } البرنامج الفرعى A
   ...                 }
160 RETURN             }
200 REM SUBROUTINE B  }
   ...                 }
225 GOSUB 100          } البرنامج الفرعى B
   ...                 }
245 RETURN             }
```

البرامج الفرعية ليست متداخلة تداخلا صحيحاً (يستدعى البرنامج الفرعى A البرنامج الفرعى B وهو بالتالى يستدعىالبرنامج الفرعى A) .

٦ – ٤٠ اكتب جزءاً من برنامج يولد رسماً بيانياً للدالة $y = \sin t$ ويولد 130 نقطة متباعدة بمسافات متضاعفة ، حيث المحور t يتجه إلىوسط الصفحة المطبوعة . دع الزيادة فى الزمن تكون بمقدار 0.1 ثانية .

```
10 FOR J=0 TO 70
20     IF J=35 THEN 50
30     PRINT TAB(J);".";
40     GO TO 60
50     PRINT TAB(J);"*";
60 NEXT J
70 PRINT
80 PRINT TAB(35);"."
90 LET T=0
100 FOR I=2 TO 130
110     LET T=T+.1
120     LET J=35+INT(35*SIN(T))
130     IF J>35 THEN 190
140     IF J=35 THEN 170
150     PRINT TAB(J);"*";TAB(35);"."
160     GO TO 200
170     PRINT TAB(J);"*"
180     GO TO 200
190     PRINT TAB(35);".";TAB(J);"*"
200     PRINT TAB(35);"."
210 NEXT I
220 END
```

مسائل تكميلية
Supplementary Problems

٦ – ٤١ اكتب دالة بيسك لحساب قيمة كل من الصيغ الجبرية المبينة فيما يلى :

(أ) $y = ax^b$

(ب) $q = c_0 + c_1r + c_2r^2 + c_3r^3 + c_4r^4$

(جـ) $i = (j+k)^{j+k}$

(د) $r = \begin{cases} \sqrt{b^2 - 4ac} & \text{if } b^2 > 4ac \\ \sqrt{4ac - b^2} & \text{if } b^2 < 4ac \end{cases}$

٦ – ٤٢ اكتب دالة بيسك لكل من المواقف الموصوفة فيما يلى :

(أ) إذا كانت Z تمثل كية موجبة يمكن أن تتعدى قيمتها الواحد الصحيح ، أوجد قيمة صحيحة مقربة .

(ب) احسب حاصل ضرب عدد N من العناصر الأولى من قائمة رقية T ، أى احسب حاصل الضرب
T(1) *T(2)*........ T(N)

(جـ) ولد 5 أرقاماً عشوائية كل منها قيمته بين A و B ، حيث A و B تمثل كيات موجبة و A < B والنتيجة هى أكبر قيمة .

(د) افحص إشارة الرقم الممثل بالمتغير X ، إذا كانت X سالبة فالنتيجة هى NEGATIVE وإذا كانت X موجبة فالنتيجة هى POSITIVE أما إذا كانت قيمة X هى صفر فالنتيجة هى ZERO .

(هـ) نفرض أن N$ تمثل كلمة متعددة الحروف . افحص كل حرف من الحروف والنتيجة هى الحرف الذى يظهر أول الحروف الهجائية .

٦ – ٤٣ كل من المواقف الموصوفة فيما يلى تتطلب استدعاء دالة معرفة فى المسألة ٦ – ٤١ أو ٦ – ٤٢ . اكتب جملة بيسك ملائمة أو مجموعة متتالية من الجمل فى كل حالة .

(أ) احسب قيمة $t = (c_1 + c_2)(x + y)^3$ (أنظر مسألة ٦ – ٤١ (أ)) .

(ب) احسب قيمة $q = c_0 + c_1 \log(x) + c_2 \log(x)^2 + c_3 \log(x)^3 + c_4 \log(x)^4$ (أنظر مسألة ٦–٤١ (ب)) .

(جـ) اطبع قيمة $f = (a - b + c)^{a-b+c}$ (أنظر مسألة ٦ – ٤١ (جـ)) .

(د) احسب الفرق بين رقم ممثل بالمتغير X وأقرب رقم صحيح له . عبر عن هذا الفرق بكية موجبة . (أنظر المسألة ٦ – ٤٢ (أ)) .

(هـ) قائمة رقية تحتوى على 61 عنصرا . مبتدئاً بالعنصر T(1) ، قرر كم عدد العناصر المتعاقبة التى يجب أن تضرب فى بعضها من أجل أن يتعدى حاصل الضرب الكية 1000 نفرض أن كل الكيات موجبة . (أنظر المسألة ٦ – ٤٢ (ب)) .

(و) ولد 20 مجموعة كل منها بها 5 أرقام عشوائية ، وكل من هذه الأرقام قيمتها تتراوح ما بين 2 و 5 . اطبع أكبر رقم عشوائي تم الحصول عليه في كل مجموعة من هذه المجموعات المكونة من 5 أرقام . (أنظر المسألة ٦ – ٤٢ (ج)) .

٦ – ٤٤ تبين كل من المسائل التالية جزءاً من برنامج بيسك يحتوى على دالة أو برنامج فرعى . يوجد على الأقل خطأ واحد فى كل حالة تعرف على كل الأخطاء .

(أ)

```
10 DEF FNK(J,K)=(C1*J+C2*K)/(J+K)
   ...
60 LET T=FNK(A,B,C)
```

(ب)

```
10 DEF FNC(X,Y)
20 IF X<Y THEN 50
30 LET C=SQR((X-Y)/2)
40 RETURN
50 LET C=SQR(X/(X+Y))
60 RETURN
70 FNEND
```

(ج)

```
 50 GOSUB 200
    ...
 80 GO TO 230
    ...
200 REM SUBROUTINE A  }
    ...               }
230 LET Z=X+Y         }  برنامج فرعى
    ...               }
250 RETURN            }
260 FNEND
```

(د)

```
10 DEF FNZ1(A↑2,B↑2)
   ...
50 LET FNZ1=(A↑2-B↑2)/(A↑2+B↑2)
60 FNEND
```

(هـ)

```
10 DEF FNK(J,K)=(C1*J+C2*K)/(J+K)
   ...
80 PRINT J,K,FNK
```

(و)

```
100 GOSUB 200
    ...
200 REM SUBROUTINE A  }
    ...               }
240 IF D<.01 THEN 150 }  برنامج فرعى
    ...               }
270 RETURN            }
```

(ز)

```
 10 DEF FNX(A,B,C)
    ...
 50 GOSUB 300
    ...
 80 FNEND
    ...
300 REM FIRST SUBROUTINE }
    ...                  }
330 LET Y=FNX(U,V,W)     }  برنامج فرعى
    ...                  }
350 RETURN               }
```

مسائل للبرمجة
Programming Problems

٦ – ٤٥ عدل البرنامج المبين فى مثال ٦ – ٦ **لإيجاد أقل قيمة** لدالة معينة .استخدم البرنامج لتحصل على جذور المعادلات التالية ، باستخدام الطريقة الموصوفة فى نهاية مثال ٦ – ٦ .

(أ) $x + \cos x = 1 + \sin x, \ \pi/2 < x < \pi$

(ب) $x^5 + 3x^2 = 10, \ 0 \leq x \leq 3$ (أنظر مثال ٤ – ٥)

٦ – ٤٦ عدل البرنامج المبين فى مثال ٦ – ١٥ مع استبدال الدالة FNP ببرنامج فرعى .

٦ – ٤٧ عدل البرنامج المبين فى مثال ٦ – ٢٠ حتى يمكن محاكاة ألعاب الكرابس المتتالية أوتوماتيكياً ، بطريقة غير تخاطبية ضمن عداد يقرر مجموع مرات الفوز ، ومتغير للمدخلات سوف تتعدد قيمته عدد مرات اللعب التى سنقوم بمحاكاتها .

استخدم البرنامج لمحاكاة عدد كبير من الألعاب (مثل ، 1000) . قدر احتمالات الفوز عند لعب الكرابس (تمثل هذه القيمة برقم عشرى مساو لعدد مرات الفوز مقسوماً على إجمالى عدد الألعاب . إذا تعدت الاحتمالات 0.500 ، فهى فى صالح اللاعب ، وإلا فهى فى غير صالحه) .

٦ – ٤٨ عدل البرنامج المبين فى مثال ٦ – ٢٦ لتشغيل المرتبات الأسبوعية . استعمل دالة بدلا من برنامج فرعى وذلك لحساب مقدار الضريبة الفيدرالية المستقطعة .

مقدار ضريبة الدخل الفيدرالية المستقطعة على أساس أسبوعى مبينة فى جدول ٦ – ٣ . هذه الأرقام مبينة على أساس إجمالى الدخل الأسبوعى المعدل ، وهو مساو لإجمالى الدخل الإسبوعى منقوصاً منه 35.58 $ لكل معول .

جدول ٦ – ٣ نسبة استقطاعات ضريبة الدخل الفيدرالية

فترة المرتبات الاسبوعية							
الاشخاص المتزوجون				شخص اعزب متضمنا ذلك رب الأسرة			
مقدار ضريبة الدخل التى سوف تستقطع هى :		اذا كان اجمالى الاجور هو :		مقدار ضريبة الدخل التى سوف تستقطع هى :		اذا كان اجمالى الاجور هو	
0		لا تزيد عن $20		0		لاتزيد عن $20	
	التى تزيد عن —	تزيد عن	ولكن ليست اكثر من :		التى تزيد عن —	تزيد عن	ولكن ليست اكثر من
14%	—$20	$20	—$42	14%	—$20	$20	—$31
$3.08 plus 17%	—$42	$42	—$77	$1.54 plus 17%	—$31	$31	—$50
$9.03 plus 16%	—$77	$77	—$163	$4.77 plus 20%	—$50	$50	—$100
$22.79 plus 19%	—$163	$163	—$269	$14.77 plus 18%	—$100	$100	—$135
$42.93 plus 21%	—$269	$269	—$385	$21.07 plus 21%	—$135	$135	—$212
$67.29 plus 25%	—$385	$385	—	$37.24 plus 24%	—$212	$212	—

سوف تحسب ضريبة الولاية كواحد فى المائة من إجمالى الدخل إذا كان فى حدود 150 $ أسبوعياً ، وواحد ونصف فى المائة لأى دخل إضافى حتى 500 $ و 2% لأى زيادة عن 500 $ سوف تحسب الضريبة المحلية على أساس 1% من أول 200 $ من إجمالى الدخل . الدخل الأسبوعى الذى يزيد عن 200 $ لن تحسب عليه ضريبة على المستوى المحلى .

٦ – ٤٩ عدل برنامج ارتداد الكرة فى المثال ٦ – ٢٨ لحل المسألة التالية . تحدد وضع الهدف على بعد مسافة L قدم من المصدر

(وتكون قيمة L كية مدخلة) . قرر بطريقة المحاولة والخطأ ما هى السرعة الأفقية التى يجب أن تأخذها الكرة من أجل تحقيق الهدف بعد الارتداد الثانى . نفرض أن الهدف هو دائرة صغيرة موضوعة على الأرض . بين موضع الهدف على المخرج البيانى .

٦ – ٥٠ اكتب برنامج بيسك يولد صورة بيانية على ورق الطباعة لكل من الدوال الآتية :

(أ) $y = 2\sqrt{x}$ لقيم x تتراوح ما بين 0 إلى 10

(ب) $y = x^3$ لقيم x تتراوح ما بين — 1 إلى + 1

(ﺟ) $y = 2e^{-0.1t} \sin 0.5t$ لقيم t تتراوح ما بين 0 إلى 60 (أنظر المسألة ٥ – ٤٨) .

وفى كل حالة ارسم نقطاً كافية للرسم حتى يرى المنحنى بطريقة واضحة .

٦ – ٥١ اكتب برنامج بيسك يرسم صورة للعلم الأمريكى . استخدم نجمة (*) للإشارة إلى كل نجمة من نجوم العلم . ومثل كل خط من خطوط العلم بعدة سطور مكررة من الحروف R أو W معتمداً فى ذلك على لون الخط .

٦ – ٥٢ جهز مخططاً تمهيدياً مفصلا وخريطة سير عمليات مناظرة ثم برنامج بيسك كامل لكل مسألة من المسائل التالية . ضمن فى البرنامج دوال وبرامج فرعية فى أى مكان تجده مناسباً .

(أ) احسب متوسط مجموعة من الأرقام عددها N . نفذ الحسابات فى البرنامج بواسطة دالة معرفة بواسطة المبرمج . ثم استخدم البرنامج لتشغيل بيانات درجات الحرارة المعطى فى المسألة ٥ – ٥٤ .

(ب) طول البرنامج المعطى فى المسألة ٦ – ٥٢ (أ) لحساب انحرافات كل نقطة عن المتوسط . استخدم البرنامج لتشغيل بيانات درجات الحرارة المعطى فى المسألة ٥ – ٥٤ . هل يمكن لدالة معرفة بواسطة المبرمج أن تستعمل لمثل هذا الفرض .

(ﺣ) احسب المساحة تحت المنحنى ، باستخدام الطرق التى تم وصفها فى المسائل ٥ – ٥٧ (د) و ٥ – ٥٧ (ك) . استخدم البرنامج لحساب المساحة تحت المنحنى $y = x^3$ بين الحدود $x = 1$ و $x = 4$

(د) طريقة أخرى لحساب المساحة تحت منحنى هى استخدام **طريقة مونت كارلو** ، وهى بالتالى تستخدم لتوليد الأرقام . العشوائية . نفرض أن المنحنى $y = f(x)$ منحنى موجب لأى قيمة من قيم x بين الحدين الأدنى والأعلى المعطيين وهما $x = a$ و $x = b$. اجعل أكبر قيمة من قيم y هى y^* وسوف نواصل فى طريقة مونت كارلو كالتالى :

(i) إبدأ بعداد قيمته الأولية صفر

(ii) ولد رقم عشوائى ، r_x ، حيث تقع قيمته بين a و b

(iii) احسب قيمة $y(r_x)$

(iv) ولد رقم عشوائى آخر ، r_y ، حيث تقع قيمته بين 0 ، $y*$

(v) قارن الرقم r_y بقيمة $y(r_x)$ إذا كانت قيمة r_y أقل من أو تساوى قيمة $y(r_x)$ ، فإن هذه النقطة سوف تقع تحت أو على المنحنى المطلوب . من ثم تزاد قيمة العداد بواحد .

(vi) تكرر الخطوات إبتداء من الخطوة (ii) إلى الخطوة (v) عدداً كبيراً من المرات . وكل مرة من هذه المرات تسمى **دورة** .

(vii) عند الانتهاء من عدد معين من الدورات ، نحسب العدد الكسرى من النقط التى تقع تحت المنحنى وتسمى F وتنتج من قسمة قيمة العداد على العدد الإجمالى للدورات . وبذلك نحصل على المساحة تحت المنحنى وهى :

$$A = Fy^* (b - a)$$

اكتب برنامج بيسك لتنفيذ هذا الإجراء . واستخدم البرنامج فى إيجاد المساحة تحت المنحنى $y = e^{-x^2}$ بين الحدود $x = 0$ و $x = 1$ قرر كم عدد الدورات المطلوبة للحصول على الإجابة الصحيحة بثلاثة أرقام معنوية .
قارن وقت الحاسب المطلوب لهذه المسألة بالوقت المطلوب فى المسألة رقم ٥ – ٥٧ (ك) .

(هـ) احسب متوسط درجات كل طالب فى الفصل الدراسى ، ثم احسب متوسط الفصل الدراسى من المتوسطات الفردية . (أنظر المسائل ٤ – ٤٨ (ز) إلى ٤ – ٤٨ (ط)) . **حدد القيمة الوسيطة** للمتوسطات الفردية (وهى قيمة مساوية أو تتعدد متصف المتوسطات الفردية) . استخدم دالة لحساب المتوسطات ، ودالة أخرى لحساب الوسيط . ثم طبق البرنامج على البيانات المعطاة فى المسألة ٤ – ٤٨ (ز) . هل يمكن استخدام برامج فرعية بدلا من الدوال ؟

(و) متغير عشوائى موزع طبيعياً x ، متوسطة μ وانحرافه المعيارى σ يمكن توليده من الصيغة التالية :

$$x = \mu + \sigma \frac{\sum_{i=1}^{N} r_i - N/2}{\sqrt{N/12}}$$

حيث r_i هى رقم عشوائى موزع بانتظام وقيمته تقع بين 0, 1 وغالباً ماتختار قيمة $N = 12$ عند استخدام هذه الصيغة . والجزء الأساسى لهذه الصيغة الذى يفهم ضمنياً هو **نظرية الحد المركزية** ، حيث تنص على أن مجموعة من قيم المتوسطات لمتغيرات عشوائية موزعة توزيعاً منتظماً سوف تكون موزعة طبيعياً .

اكتب برنامج بيسك يولد أرقاماً معطاة لمتغيرات عشوائية موزعة طبيعياً بمتوسطات وانحرافات معيارية معطاة . اجعل عدد المتغيرات العشوائية والمتوسطات والانحرافات المعيارية كلها معاملات إدخال .

استخدم البرنامج لتوليد هيستوجرام للتوزيع الطبيعى حيث $\mu = 2.5$ و $\sigma = 1.5$

(ز) اكتب برنامج بيسك يسمح لشخص أن يلعب لعبة تيك – تاك – تو ضد الحاسب . اكتب البرنامج بطريقة يمكن أن يكون فيها الحاسب أما اللاعب الأول أو اللاعب الثانى . إذا كان الحاسب هو اللاعب الأول ، فدع الحركة الأولى تولد عشوائياً . اكتب حالة اللعبة كاملة بعد كل حركة . اجعل الحاسب يتعرف على الطرف الفائز عند حدوث ذلك .

(ح) اكتب برنامج بيسك يحاكى لعبة (بلاك جاك) بين لاعبين ولاحظ أن الحاسب لن يشارك فى هذه اللعبة كلاعب ولكنه ببساطة سوف يوزع الورق لكل لاعب ويعطى كل لاعب « ضربة » أو أكثر (وذلك يعنى كروتا إضافية) عند الطلب .

وتوزع الكروت بالترتيب ، أولا كرت لكل لاعب ثم كرت آخر لكل لاعب ويمكن أن تطلب ضربات إضافية بعد ذلك .

والهدف من اللعبة هو الحصول على 21 نقطة أو أى عدد ممكن من النقط ولكن لا تتعدى 21 نقطة فى كل يد . ويصبح اللاعب غير مؤهل أوتوماتيكياً إذا تعدت النقط بيده عن 21 نقطة . وتحسب الصور على أساس 10 نقط ، ويمكن أن يحسب كرت الواحد بنقطة أو 11 نقطة . وبذلك يمكن أن يحصل اللاعب على 21 نقطة من أول كرتين (بلاك جاك !) إذا كان الكرتين اللذين تم توزيعهما أحدهما واحد والأخرى أما عشرة أو صورة . وإذا كانت عدد النقط التى تم الحصول عليها من أول كرتين قليلة فيمكن أن يطلب اللاعب بعد ذلك ضربة أو أكثر .

يجب أن يستخدم مولد الرقم العشوائى ليحاكى توزيع الكروت . وتأكد من تضمين شرط عدم توزيع نفس الكرت أكثر من مرة .

(ط) لعبة الروليت تلعب بعجلة تحتوى على 38 مربعاً مختلفاً حول محيطها . بينهما مربعان ، أرقامهما 0و00 ، لونهما أخضر ، 18 مربعاً لونها أحمر ، 18 مربعاً لونها أسود . وتتعاقب المربعات السوداء والحمراء وترقم من 1 إلى 36 بترتيب عشوائى .

تدار بلية صغيرة بداخل العجلة ، والتى سوف تستقر فى آخر الأمر فى حفرة وراء أحد المربعات . وتلعب اللعبة بالرهان على نتيجة الدوران ، بأى طريقة من الطرق التالية :

١ – بانتقاء مربع واحد أحمر أو أسود ، مقابل 35 إلى 1 (وبذلك إذا راهن اللاعب بدولار واحد وكسب الرهان ، فسوف يتسلم مبلغاً إجمالياً وقدره 36.00$ – الدولار الأصلى علاوة على 35.00 $ إضافى) .

٢ – بانتقاء لون (أما أحمر أو أسود) مقابل 1 إلى 1 (وبذلك إذا اختار اللاعب اللون الأحمر وراهن بدولار واحد ، فسوف يتسلم 2.00 $ إذا استقرت البلية وراء أى مربع أحمر) .

٣ – بانتقاء أما الأرقام الفردية أو الزوجية (باستبعاد 0 و 00) مقابل واحد لواحد .

٤ – بانتقاء أما 18 رقماً السفلية أو18 رقما العليا مقابل واحد لواحد .

وسوف يخسر اللاعب أوتوماتيكياً إذا استقرت البلية وراء أى من المربعات الخضراء (0 أو 00) .

اكتب برنامج بيسك من النوع التخاطبى الذى يحاكى لعبة الروليت . واسمح للاعب أن ينتقى أى نوع من اللعب حسب رغبته . ثم اطبع نتيجة كل لعبة تتبعها رسالة مناسبة تشير إلى خسارة اللاعب أو مكسبه .

(ى) اكتب برنامج بيسك يكود أو يفك شفرة سطر من النص (سلسلة حرفية) ولتكويد السطر اتبع مايلى :

١ – تحويل أى حرف (حتى الأماكن الخالية) لما يقابله من كود ASCII .

٢ – ولد رقم عشوائى صحيح موجب . أضف هذا الرقم الصحيح إلى قيمة كود ASCII المكافئة لكل حرف . (سوف يستخدم نفس الرقم الصحيح فى السطر الكامل) .

٣ – إفرض أن N1 تمثل أقل قيمة مسموح بها فى كود ASCII وN2 تمثل أعلى قيمة مسموح بها . إذا تعدت قيمة الرقم الذى تم الحصول عليها فى الخطوة ٢ العالية قيمة N2 (أى الرقم الأصلى المقابل لكود ASCII مضافاً إليه ومن الرقم العشوائى) ، فاطرح أكبر رقم مسموح به لمضاعفات N2 من هذا الرقم ، ثم أضف الباقى للرقم N1 ثم فإن الرقم الذى يكود يقع بين N1 و N2 وعلى ذلك تمثل دائماً بعض الحروف .

٤ – اطبع الأحرف التى تناظرها حسب كود ASCII .

سوف يعكس الإجراء فى حالة فك شفرة سطر من النص . ومع ذلك ، تأكد ، أن نفس الرقم العشوائى هو المستخدم فى حالة التكويد وفك الشفرة .

(ك) اكتب برنامج بيسك . يحاكى لعبة (بنجو) BINGO اطبع كل توافقية حرف – رقم كما يتم سحبها . تأكد من أن التوافقية لاتسحب إلا مرة واحدة تذكر أن كلا من الحروف B-I-N-G-O تناظر مدى معينا من الأرقام ، كما يلى :

B:	1–15
I:	16–30
N:	31–45
G:	46–60
O:	61–75

الفصـل ٧

المتجهات والمصفوفات
Vectors and Matrices

تعلمنا فى الفصل الخامس أن كل عناصر المجموعة المتراصة يمكن أن يشار إليها جميعاً بإعطاء اسم مجموعة متراصة مشترك . بينما يلزم عند التعامل مع المجموعات المتراصة أن نتعامل مع كل عنصر من عناصر المجموعة المتراصة على حدة (أى كل متغير ذى دليل) . وغالباً ما يتم إنجاز ذلك بواسطة الحلقة التكرارية FOR-TO

تحتوى معظم نسخ البيسك على مجموعة خاصة من الجمل . تعرف باسم **جمل المصفوفات** . وذلك للقيام بعمليات المجموعات المتراصة الأكثر شيوعاً . وغالباً سوف تستخدم جملة مصفوفات واحدة لعملية معينة . وبذلك يمكن تنفيذ عملية معينة على كل عناصر المجموعة المتراصة بدون استخدام حلقة FOR-TO . فى هذا الفصل سوف نرى كيف يمكننا إنجاز ذلك .

٧ — ١ عمليات المتجهات والمصفوفات VECTOR AND MATRIX OPERATIONS

« متجه » و « مصفوفة » مصطلحات رياضية تشير إلى قائمة أو إلى جدول على الترتيب . وبذلك يكون المتجه هو مجموعة متراصة لها بعد واحد والمصفوفة هى مجموعة متراصة ذات بعدين وحيث أن المتجه هو حالة خاصة من المصفوفة ، فإن معظم القواعد العامة التى تطبق على المصفوفات يمكن تطبيقها على المتجهات .

وكما ذكرنا سابقاً ، فسوف نستخدم متغيرات ذات أدلة لتمثل كل عنصر من عناصر المجموعة المتراصة على حدة . وفى حالة المصفوفة سوف نجعل الدليل الأول يشير إلى الصف والدليل الثانى يشير إلى العمود . وبذلك فإن A (3, 2) سوف تمثل العنصر فى الصف الثالث والعمود الثانى من المصفوفة A . وعلاوة على ذلك فسوف نشير إلى مصفوفة لها عدد m من الصفوف و n من الأعمدة كمصفوفة $m \times n$ (تذكر أن حجم المتجه أو المصفوفة يجب أن يحدد بواسطة جملة DIM إذا تعدت قيمة الدليل الرقم 10) .

أغلب عمليات المتجهات والمصفوفات الشائعة هى الجمع والطرح والضرب اللاموجه والضرب الموجه . ولكل من هذه العمليات جملة بيسك خاصة . وتحتوى اللغة أيضاً على جملة تحديد قيم للمصفوفة . وسوف تتم مناقشة كل جملة من هذه الجمل على حدة فيما بعد .

تحـديد قيمـة Assignment

جملة تحديد قيم للمصفوفة تكون على النحو التالى :

```
10  MAT  C = A
```

يترتب على هذه الجملة تحديد قيمة لكل عنصر من عناصر المصفوفة C بقيمة العنصر المناظر من المصفوفة A .

مثال ٧ – ١

نفرض أن المصفوفة A تمثل المصفوفة (3 × 2) التالية :

$$A = \begin{bmatrix} 3 & 5 & -9 \\ 2 & -6 & 7 \end{bmatrix}$$

وجملة المصفوفة

```
10 MAT C=A
```

سوف يترتب عليها جمل عناصر المصفوفة (3 × 2) التى تسمى C كالتالى :

$$C = \begin{bmatrix} 3 & 5 & -9 \\ 2 & -6 & 7 \end{bmatrix}$$

تستعمل المتغيرات ذات الأدلة للإشارة إلى كل عنصر من عناصر المصفوفة على حدة وبذلك فإن :
C (1, 1) = 3 ، C (1, 2) = 5 ، C (1, 3) = 9 ، C (2, 1) = 2 ، C (2, 2) = — 6 ، C (2, 3) = 7

الجمع Addition

تتم عملية جمع المصفوفات بجملة تكتب على النحو التالى :

```
10 MAT C=A+B
```

نتيجة هذه الجملة هو تحديد كل عنصر من عناصر المصفوفة C بحاصل جمع العنصرين المناظرين فى المصفوفتين A ، B أى C (I, J) = A (I, J) + B (I, J) . لابد أن تكون المصفوفتان A و B لهما نفس عدد الصفوف و نفس عدد الأعمدة .

مثال ٧ – ٢

نفرض أن كلا من المصفوفتين A و B (3 × 2) وقيم عناصرها كما يلى :

$$A = \begin{bmatrix} 3 & 5 & -9 \\ 2 & -6 & 7 \end{bmatrix} \qquad B = \begin{bmatrix} 2 & 2 & 0 \\ -4 & 5 & 1 \end{bmatrix}$$

وسوف تسبب الجملة :

```
10 MAT C=A+B
```

فى جمل عناصر المصفوفة (3 × 2) C كما يلى :

$$C = \begin{bmatrix} (3+2) & (5+2) & (-9+0) \\ (2-4) & (-6+5) & (7+1) \end{bmatrix} = \begin{bmatrix} 5 & 7 & -9 \\ -2 & -1 & 8 \end{bmatrix}$$

يمكن أن تعدل القيم المعطاة لمصفوفة ما بجملة الجمع ، مثال لذلك جملة الجمع بالصورة التالية :

```
10 MAT A=A+B
```

مسموح بها . بينما جملة الجمع المتعددة مثل :

```
10 MAT D=A+B+C
```

غير مسموح بها .

الطرح Subtraction

جملة طرح المصفوفة مشابه لجملة جمع المصفوفة تماماً فيما عدا استبدال علامة الجمع (+) بعلامة الطرح (−) ، وبذلك فالجملة :

```
10 MAT C=A-B
```

سوف تتسبب فى إعطاء قيم لعناصر المصفوفة C مساوية لحاصل طرح القيمتين المناظرتين فى A و B أى C (I, J)=A (I, J)−B (I, J) ويجب أن يكون للمصفوفتين A و B نفس عدد الأعمدة ونفس عدد الصفوف .

مثال ٧ − ٣

نفرض أن A و B مصفوفتان (3 × 2) ولها نفس العناصر كما فى مثال ٧ − ٢ . فبجملة المصفوفة :

```
10 MAT C=A-B
```

سوف يترتب عليها إعطاء عناصر المصفوفة C القيم التالية :

$$C = \begin{bmatrix} (3-2) & (5-2) & (-9-0) \\ (2+4) & (-6-5) & (7-1) \end{bmatrix} = \begin{bmatrix} 1 & 3 & -9 \\ 6 & -11 & 6 \end{bmatrix}$$

وكما ذكرنا فى جملة جمع المصفوفات ، فيمكن تعديل القيم المعطاة للمصفوفة من خلال جملة الطرح . وبذلك فإن جملة مثل :

```
10 MAT A=A-B
```

مسموح بها . فى حين أن جمل المصفوفات التالية :

```
10 MAT D=A-B-C
```

و

```
10 MAT D=A+B-C
```

غير مسموح بهما .

الضرب غير الموجه Scalar Multiplication

فى الضرب غير الموجه نضرب كل عنصر من عناصر المصفوفة ، برقم ثابت . ويمكن إنجاز ذلك فى البيسك بواسطة الجملة التالية :

```
10 MAT C=(K)*A
```

حيث A و C مصفوفتان و K متغير عادى . وسوف يكون كل عنصر من عناصر C له القيمة C (I, J) = (K)*A (I, J)

مثال ٧ – ٤

نفرض أن المصفوفة A هى نفس المصفوفة (3 × 2) المعطاة فى الأمثلة ٧–١ ، ٧ – ٢ ، وأن K متغير له القيمة 3.5 فإن الجملة :

```
10 MAT C=(K)*A
```

سوف يترتب عليها إعطاء عناصر المصفوفة C القيم التالية :

$$C = (3.5)*\begin{bmatrix} 3 & 5 & -9 \\ 2 & -6 & 7 \end{bmatrix} = \begin{bmatrix} 10.5 & 17.5 & -31.5 \\ 7 & -21 & 24.5 \end{bmatrix}$$

لا يستلزم أن يكون الحد الموجود بين القوسين مجرد متغير . ولكن يمكن أن يظهر بين القوسين أيضاً ثوابت أو متغيرات ذات أدلة أو صيغ رياضية أو إشارة إلى دوال . ولكن الشرط هو ؛ أن هذا الحد يجب أن يمثل كمية رقية . وهذا الحد العددى يجب أن يكون محصوراً بين قوسين .

مثال ٧ – ٥

مبين فيما يلى عدة أمثلة صحيحة لجمل الضرب غير الموجه :

```
10 MAT C=(100)*A
10 MAT C=(2*X+Y)*A
10 MAT C=(SQR(P↑2+Q↑2))*A
```

وفى هذه الأمثلة A و B مصفوفتان وX و Y و P و Q متغيرات عددية عادية و SQR تمثل الدالة المكتبية للجذر التربيعى .

يمكن أن تعدل قيمة المتجه بواسطة جملة الضرب غير الموجه . وبذلك فإن الجملة :

```
10 MAT A=(10)*A
```

مسموح بها . وذلك كما حدث فى جمع المصفوفة وطرح المصفوفة . بينما جملتين مثل

```
10 MAT C=(10)*A*B
```

و

```
10 MAT C=(10)*A+B
```

غير مسموح بهما .

ضرب المصفوفات Matrix Multiplication

يمكن ضرب مصفوفتين إذا كان عدد أعمدة المصفوفة الأولى مساوياً لعدد صفوف المصفوفة الثانية . والنتيجة سوف تكون مصفوفة لها نفس عدد صفوف المصفوفة الأولى ونفس عدد أعمدة المصفوفة الثانية . وبذلك فإذا كانت A هى مصفوفة $(k \times m)$ و B مصفوفة $(m \times n)$ فإن جملة المصفوفة C = A*B سوف تولد مصفوفة C لها عدد صفوف مساو k وعدد أعمدة مساو n وكل عنصر من عناصر المصفوفة C سوف نحصل عليه نتيجة للعملية :

```
C(I,J)=A(I,1)*B(1,J)+A(I,2)*B(2,J)+···+A(I,K)*B(K,J)
```

ويمكن القيام بعملية ضرب المصفوفات فى البيسك بواسطة الصيغة :

```
10 MAT C=A*B
```

حيث كل من A و B و C مصفوفات ، حيث A لها عدد من الأعمدة مساو لعدد الصفوف الموجودة فى B .

مثال ٧ – ٦

نفرض أن لدينا المصفوفتين التاليتين :

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \end{bmatrix} \qquad B = \begin{bmatrix} 9 & 5 & 1 \\ 8 & 4 & 0 \\ 7 & 3 & 9 \\ 6 & 2 & 8 \end{bmatrix}$$

فإن جملة المصفوفة :

```
10 MAT C=A*B
```

سوف تنتج مصفوفة $C\ (2\times3)$

$$C = \begin{bmatrix} 70 & 30 & 60 \\ 190 & 86 & 132 \end{bmatrix}$$

حيث يمكن الحصول على كل عنصر من عناصر المصفوفة كما يلى

$C(1,1)=(1\times9)+(2\times8)+(3\times7)+(4\times6)=70$
$C(1,2)=(1\times5)+(2\times4)+(3\times3)+(4\times2)=30$
$C(1,3)=(1\times1)+(2\times0)+(3\times9)+(4\times8)=60$
$C(2,1)=(5\times9)+(6\times8)+(7\times7)+(8\times6)=190$
$C(2,2)=(5\times5)+(6\times4)+(7\times3)+(8\times2)=86$
$C(2,3)=(5\times1)+(6\times0)+(7\times9)+(8\times8)=132$

ولكن بعكس جمل المصفوفات التى تم ذكرها سابقاً فإن المصفوفة لا يمكن تعديل قيمها بواسطة جملة ضرب المصفوفة . ولا يمكن أيضاً إجراء عملية الضرب على أكثر من مصفوفتين فى وقت واحد . وبذلك فإن الجملتين التاليتين :

```
10 MAT A=A*C
```

و

```
10 MAT D=A*B*C
```

غير مسموح بهما ، بينما يمكن ضرب مصفوفة فى نفسها ، أى ، يمكن كتابة :

```
10 MAT C=A*A
```

بشرط أن تكون المصفوفة A مصفوفة مربعة (أى يجب أن يكون لها عدد من الصفوف مساوياً لعدد الأعمدة) .

٧ – ٢ ادخال / اخراج المتجهات والمصفوفات VECTOR AND MATRIX INPUT/OUTPUT

عمليات إدخال وإخراج المصفوفات يمكن القيام بها بنفس الطريقة التى تتم بها العمليات العادية للإدخال / الإخراج . يمدنا البيسك بثلاث جمل (للإدخال والإخراج) I/O من جمل المصفوفات وهى *MAT READ* و *MAT PRINT* ، *MAT INPUT* وسوف نناقش كلا على حدة فيما يلى .

MAT READ

الغرض من جملة MAT READ هو إدخال قيم لعناصر المتجه أو المصفوفة . تستخدم هذه الجملة مقترنة بجملة أو أكثر من جمل DATA (انظر القسم ٥ . ٥) . يمكن أن تظهر جملة MAT READ النموذجية كما يلي :

```
10 MAT READ A
```

حيث تمثل A متجهاً أو مصفوفة لها أبعاد سبق تحديدها .

يترتب على تنفيذ جملة MAT READ إعطاء القيم الموجودة فى كتلة البيانات لعناصر المتجه أو المصفوفة المناسبة . يبدأ التحديد دائماً بالدليل (أو الأدلة) المساوية للواحد الصحيح ، أى أن العنصر الذى رقمه صفر يتم تجاهله . فى حالة المصفوفة ، يتم تحديد البيانات على أساس صف ثم الصف التالى .

مثال ٧ – ٧

جزء من برنامج بيسك مبين فيما يلى :

```
 10 DIM A(5,3)
    ...
 40 MAT READ A
    ...
200 DATA 1,3,5,7,9,11,13,15,17,19,21,23,25,27,29
```

ويترتب على تنفيذ هذا البرنامج تحديد عناصر المصفوفة A بالقيم التالية :

A(1,1)=1	A(1,2)=3	A(1,3)=5
A(2,1)=7	A(2,2)=9	A(2,3)=11
A(3,1)=13	A(3,2)=15	A(3,3)=17
A(4,1)=19	A(4,2)=21	A(4,3)=23
A(5,1)=25	A(5,2)=27	A(5,3)=29

لاحظ أن البيانات تم تحديدها صفاً بصف .

لاحظ أن المصفوفة A تحتوى حقيقة على 24 عنصراً ، حيث أن الأدلة تتراوح ما بين 0 و 5 ومن 0 إلى 3 على الترتيب . ولكن 15 عنصراً فقط من A هى التى تحدد بقيم ، بينما يتم تجاهل العناصر ذات الدليل صفر .

إذا لم تكتب جملة DIM فإن A تتكون أوتوماتيكياً من 121 عنصراً (كل دليل يتراوح من 0 إلى 10) . وبذلك يكون المطلوب 100 قيمة فى كتلة البيانات . وسوف يترتب على إعطاء 15 قيمة فقط ، كما فى جملة ATA السابقة ، حدوث خطأ .

يمكن أن تحتوى جملة MAT READ واحدة على عدة متجهات ومصفوفات إذا أردنا ذلك . يجب فصل المتجهات والمصفوفات المتعاقبة بواسطة فاصلة (,) وسوف تقرأ كل عناصر المتجه أو المصفوفة الأولى بالكامل قبل أى عنصر من عناصر المتجه أو المصفوفة الثانية و ... وهكذا . وكما ذكرنا سابقاً سوف تحدد قيم المصفوفة صفاً بصف .

مثال ٧ – ٨

جزء من برنامج بيسك ممثل فيما يلي :

```
 10 DIM X(2,2),Y(5),Z(2,3)
 50 MAT READ X,Y,Z
150 DATA 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15
```

وسوف يترتب على تنفيذ هذا البرنامج تحديد عناصر X و Y وZ بالقيم التالية :

```
X(1,1)=1        Y(1)=5        Z(1,1)=10
X(1,2)=2        Y(2)=6        Z(1,2)=11
X(2,1)=3        Y(3)=7        Z(1,3)=12
X(2,2)=4        Y(4)=8        Z(2,1)=13
                Y(5)=9        Z(2,2)=14
                              Z(2,3)=15
```

وبذلك نرى أن القيم الأربع الأولى الموجودة فى كتلة البيانات تعطى للمتجه X والقيم الخمس التالية سوف تعطى للمتجه Y والقيم الست الأخيرة للمصفوفة Z . وتحدد قيم المصفوفات صفاً بصف .

MAT PRINT

تستخدم جملة MAT PRINT لطباعة عناصر المتجه أو المصفوفة . يمكن كتابة جملة MAT PRINT نموذجية كما يلى :

```
10 MAT PRINT A
```

حيث تمثل A إما متجهاً أو مصفوفة ، وسوف تطبع عناصر A بشكل عمودى إذا كانت A متجهاً . وبشكل جدولى . أو بصورة صف بصف إذا كانت A مصفوفة . وكما فى جملة MAT READ سوف يتم تجاهل العناصر التى تحمل الدليل صفراً .

سوف تفصل عناصر كل صف من المصفوفة عن بعضها ، بحد أقصى 5 عناصر فى كل سطر مطبوع . من ثم فيمكن أن يكون المطلوب عدة سطور لكل صف . وسوف يطبع سطر خال بين السطور المتعاقبة ، وبذلك نستطيع التمييز بين صف وآخر .

مثال ٧ – ٩

ادرس برنامج البيسك التالى :

```
10 DIM X(3,8),Y(6)
20 MAT READ X,Y
30 MAT PRINT X
40 MAT PRINT Y
50 DATA 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15
60 DATA 16,17,18,19,20,21,22,23,24,25,26,27,28,29,30
70 END
```

سوف ينتج الخرج التالى عند تنفيذ هذا البرنامج :

```
1         2         3         4         5
6         7         8

9         10        11        12        13
14        15        16

17        18        19        20        21
22        23        24


25
26
27
28
29
30
```

لاحظ أن كل صف من X يتطلب سطرين ، حيث يمكن طباعة 5 عناصر فقط فى كل سطر . يفصل بين كل صفين متعاقبين سطر خال . ولاحظ كذلك أن Y تطبع على شكل عمودى حيث أنها متجه .

يمكن تعديل التباعد بين عناصر المجموعة المتراصة المتعاقبة بوضع فاصلة (,) أو فاصلة منقوطة (;) بعد ذكر اسم المجموعة المتراصة فى جملة MAT PRINT ويتم تداول المتجهات بصورة تختلف عن المصفوفات . وفيما يلى القواعد التى تحكم التباعد بين عناصر المجموعة المتراصة :

١ – المتجهات : Vectors

(ا) إذا تبع اسم المتجه فاصلة (,) تطبع العناصر فى شكل صف بدلا من شكل عمودى وسوف تكون المسافات بين العناصر متباعدة (لا يزيد عن 5 عناصر فى كل سطر) .

(ب) إذا تبع اسم المتجه فاصلة منقوطة (;) فسوف تطبع العناصر فى شكل صف مع ترك أقل مسافة بينها .

٢ – المصفوفات : Matrices

إذا تبع اسم المصفوفة فصلة (و) فلن يؤثر ذلك فى التباعد بين عناصر المصفوفة ولا فى شكل الخرج بينما إذا تبع اسم المصفوفة فاصلة منقوطة (;) فسوف تطبع المصفوفة صفاً بصف مع ترك أقل مسافة بين العناصر وبعضها . وسوف يتم فصل الصفوف بسطر خال .

مثال ٧ – ١٠

دعنا ندرس برنامج البيسك المعروض فى مثال ٧ – ٩ مرة أخرى . إذا تم تغيير جمل MAT PRINT إلى

```
30 MAT PRINT X,
40 MAT PRINT Y,
```

فسوف يتم توليد الخرج التالى عند تنفيذ البرنامج :

```
1          2          3          4          5
6          7          8

9          10         11         12         13
14         15         16

17         18         19         20         21
22         23         24


25         26         27         28         29
30
```

نرى أن المصفوفة X تم طباعتها بنفس الطريقة السابقة ولكن المتجه Y يظهر الآن على شكل صف .

ومن وجهة نظر أخرى ، نفرض أننا استبدلنا الفصلات (,) فى جملة MAT PRINT بالفصلات المنقوطة (;) ، أى . دعنا نكتب :

```
30 MAT PRINT X;
40 MAT PRINT Y;
```

فسوف يظهر الخرج كما يلى :

```
1  2  3  4  5  6  7  8
9  10  11  12  13  14  15  16
17  18  19  20  21  22  23  24


25  26  27  28  29  30
```

وسوف يتم فصل العناصر بأقل مسافة ممكنة وتظهر متقاربة .

يمكن ظهور عدة متجهات ومصفوفات فى نفس جملة MAT PRINT إذا تطلب الأمر ذلك . ويجب أن تفصل الأسماء المتعاقبة إما بواسطة فاصلة (,) أو فاصلة منقوطة (;) وسوف يحدد شكل كل متجه أو مصفوفة تبعاً لنوع علامة الفصل التى تتبع اسم المتجه أو المصفوفة .

مثال ٧ – ١١

جملتا MAT PRINT فى مثال ٧ – ٩ يمكن أن نستبدلهما بجملة واحدة .

```
30 MAT PRINT X,Y
```

وعند تنفيذ البرنامج فسوف تحدد مسافات الخرج كما هو مبين فى مثال ٧ – ٩ .

ومن وجهة نظر أخرى ، نفرض جملة MAT PRINT هذه قد تم تحويلها إلى :

```
30 MAT PRINT X,Y
```

(لاحظ إضافة الفاصلة الأخيرة) وعند تنفيذ هذا البرنامج سوف تولد مجموعة من المخرجات بينها مسافات متباعدة ، وسوف يتم عرض عناصر المتجه فى شكل صف كما هو مبين فى بداية المثال ٧ – ١٠ .

إذا استبدلت الفاصلة (,) بالفاصلة المنقوطة (;) ، أى .

```
30 MAT PRINT X;Y;
```

فإن الخرج سوف يظهر بصورة أكثر تلاصقاً ، كما هو مبين فى الجزء التالى من مثال ٧ – ١٠ .

وأخيراً ، يجب أن نشير إلى أن جملة MAT PRINT يمكن أن تحتوى على أسماء متجهات أو مصفوفات فقط . أما الصيغ الرياضية أو مراجع الدوال ، ....إلخ فغير مسموح بها . وبذلك فإن جملة على الصورة :

```
100 MAT PRINT A+B,C*D,(K)*X
```

غير مسموح بها .

MAT INPUT

تستخدم جملة MAT INPUT لإدخال عناصر متجه من النهاية الطرفية مباشرة . ويمكن أن تظهر جملة MAT INPUT النموذجية كالتالى :

```
10 MAT INPUT A
```

حيث تمثل A اسم متجه . وتسمح معظم نسخ البيسك بظهور اسم متجه واحد فقط فى جملة MAT INPUT

عند تنفيذ جملة MAT INPUT سوف تظهر علامة الاستفهام (؟) عند بداية سطر جديد ، مشيرة إلى طلب البيانات . سوف يتوقف تنفيذ البرنامج مؤقتاً أثناء إدخال عناصر المتجه المطلوبة تفصل بينها فصلات (,) . أول قيمة للبيانات سوف تعطى للعنصر A (1) ، والثانية للعنصر A (2) ... وهكذا (حيث A هو اسم المتجه) . وسوف يتم تجاهل العنصر رقم صفر .

بعد الانتهاء من إدخال آخر عنصر ، يجب على المستفيد أن يضغط على مفتاح RETURN ، وبذلك يتسبب فى نقل البيانات إلى الحاسب . بعد ذلك يتم مواصلة تنفيذ البرنامج .

من المهم أن نلحظ أنه من الممكن إدخال **أى عدد** من البيانات (على أن يؤخذ فى الاعتبار أن عدد قيم البيانات لا تتعدى الحد الأقصى المسموح به لعناصر المتجه ، والمحدد فى جملة DIM) . ومن ثم يمكن إدخال **مجموعة جزئية فقط** من العناصر من خلال جملة MAT INPUT وهذه الخاصية تعتمد على مدى استعمال جملة MAT INPUT فى عديد من برامج بيسك .

مثال ٧ – ١٢

يحتوى برنامج بيسك على الجملتين التاليتين :

```
10 DIM A(100)
   ...
50 MAT INPUT A
```

عند مقابلة الجملة رقم 50 أثناء تنفيذ البرنامج ، سوف تطبع علامة استفهام (؟) عند بداية سطر جديد . وسوف يتم تعليق تنفيذ البرنامج مؤقتاً .

نفرض أن السطر التالى من البيانات قد تم إدخاله نتيجة الإجابة على علامة الاستفهام :

```
?12,-3,17,10,62,-87,49,5,39,9,-7,-22
```

وسوف يتم نقل البيانات للحاسب بعد الضغط على مفتاح RETURN مسبباً تحديد عناصر المتجه A بالقيم التالية :

A(1)=12	A(5)=62	A(9)=39
A(2)=-3	A(6)=-87	A(10)=9
A(3)=17	A(7)=49	A(11)=-7
A(4)=10	A(8)=5	A(12)=-22

لاحظ أن عنصر المتجه A رقم (0) أى A(0) والعناصر من A (13) إلى A (100) لا تتأثر .

فى بعض الأحيان يكون المطلوب قيم بيانات عديدة على سطر واحد من النهاية الطرفية . عند حدوث ذلك يمكن إدخال قيم البيانات على أسطر متتالية . يجب أن تستخدم علامة & لتشير أن سطراً ثانياً من البيانات سوف يتم إدخاله . يجب أن تظهر (علامة &) بعد آخر قيمة فى كل سطر ما عدا آخر سطر . وسوف تطبع علامة استفهام (؟) عند بداية كل سطر .

مثال ٧ – ١٣

نفرض أن السطرين التاليين من بيانات الإدخال تم طباعتهما استجابة لجملة MAT INPUT المبينة فى مثال ٧ – ١٢ :

```
?3,6,9,12,15,18,21,24,27,30&
?33,36,39,42,45,48,51,54,57,60
```

علامة الاستفهام فى بداية السطر الثانى ، تشير لطلب مزيد من البيانات وقد تم توليدها بواسطة علامة (&) فى نهاية السطر الأول . وكان يمكن استكمال هذا الإجراء عند الضرورة (أى ، كان يمكن طباعة (&) فى نهاية السطر الثانى ، مولداً طلباً لسطر ثالث من البيانات ، ... وهكذا) .

سوف يستأنف تنفيذ البرنامج بعد الضغط للمرة الثانية لعودة العربة (وذلك بعد السطر الثانى من البيانات) . وسوف يعطى المتجه A ذو المائة عنصر قيماً لـ 20 عنصراً الأولى A(1) = 1 و A (2) = 6 و A(3) = 9 و .... ,A (19) = 57 و A(20) = 60 ولن تتأثر باقى عناصر المتجه A .

وفى بعض الأحيان يستحب أن نعرف كم عدد القيم التى تم إدخالها من النهاية الطرفية . تسمح الدالة المكتبية NUM بالإجابة على هذا السؤال . فأينما تستخدم NUM كرجع للدالة سوف تعطينا عدد قيم البيانات التى تم إدخالها فى جملة MAT INPUT الأكثر حداثة . لا تتطلب هذه الدالة أى خلاصات .

مثال ٧ – ١٤

مبين فيما يلى جزء من برنامج بيسك :

```
 10 DIM A(100)
    ...
 50 MAT INPUT A
 60 LET A(0)=NUM
    ...
150 PRINT "THE LIST CONTAINS"; A(0); " VALUES"
```

عند تنفيذ هذا البرنامج سوف يتم إدخال عدد غير محدد من قيم البيانات ويتم إعطاؤها لعناصر المتجه A. تتسبب جملة 60 فى تحديد عدد قيم البيانات وتخزينها فى العنصر رقم صفر من المتجه A .ويترتب على تنفيذ الجملة رقم 150 طباعة رسالة تشير إلى الحجم الحقيق للمتجه A.

نفرض مثلا ، أنه تم إعطاء 20 قيمة للمتجه A (كما فى مثال ٧ – ١٣) فإن الجملة سوف تولد الرسالة التالية :

تحتوى القائمة على 20 قيمة THE LIST CONTAINS 20 VALUES

حيث تم تحديد قيمة 20 للمنصر رقم (0) A فى الجملة 60 .

يمكن أن تستخدم جملة MAT INPUT فى بعض نسخ البيسك لإدخال قيم عناصر مصفوفة تماماً كعناصر المتجه . ولكن فى المصفوفة يجب أن تحدد العناصر التى يجب إدخالها من خلال جملة MAT INPUT .(وسوف نرى كيف يتم إنجاز ذلك فى قسم ٧–٤). وبذلك تصبح جملة MAT INPUT أقل نفعاً عند إدخال عناصر المصفوفة عنها للمتجه ، حيث لايمكن إدخال **مجموعة جزئية** من عناصر المصفوفة . ولهذا السبب ، ولأن معظم نسخ بيسك لا تسمح إطلاقاً باستخدام جملة MAT INPUT مع المصفوفة ، فسوف لا نناقش هذا الموضوع بعد ذلك .

يوضح المثال التالى استخدام عدة جمل مصفوفات فى برنامج بيسك كامل من بينها MAT READ و MAT INPUT وسوف نرى برنامج بيسك كاملا يستخدم جملة MAT INPUT فى جزء لاحق من هذا الفصل ، فى المثال ٧ – ٢٢ .

مثال ٧ – ١٥ مناولة المصفوفة Matrix Manipulation

نفرض أن المصفوفتين A و B كل منها (3 × 3) وتأخذ عناصرها القيم التالية :

$$A = \begin{bmatrix} 1 & 3 & 5 \\ 7 & 9 & 11 \\ 13 & 15 & 17 \end{bmatrix} \qquad B = \begin{bmatrix} 2 & 4 & 6 \\ 8 & 10 & 12 \\ 14 & 16 & 18 \end{bmatrix}$$

ونرغب فى حساب قيمة صيغة المصفوفة الرياضية التالية :

```
F=5*(A+B)*(A-B)
```

يمكن إنجاز ذلك بسهولة بواسطة جمل المصفوفة التى تم عرضها قبل ذلك فى هذا الفصل . (ويمكن حل هذه المسألة أيضاً بدون استخدام جمل المصفوفات ، بالرغم أن البرنامج سوف يصبح أكثر تعقيداً) .

طريقة إجراء الحسابات Computational Procedure

حيث أن الصيغة المعطاة لا يمكن حسابها بجملة مصفوفة واحدة ، فيجب تكوين عدد متتالٍ من عمليات المصفوفات البسيطة والتى سوف تعطينا النتيجة المطلوبة . ويمكن إنجاز ذلك كالتالى :

```
C=A+B
D=A-B
E=C*D
F=(5)*E
```

يمكن القيام بكل عملية من العمليات السابقة بجملة مصفوفة واحدة .

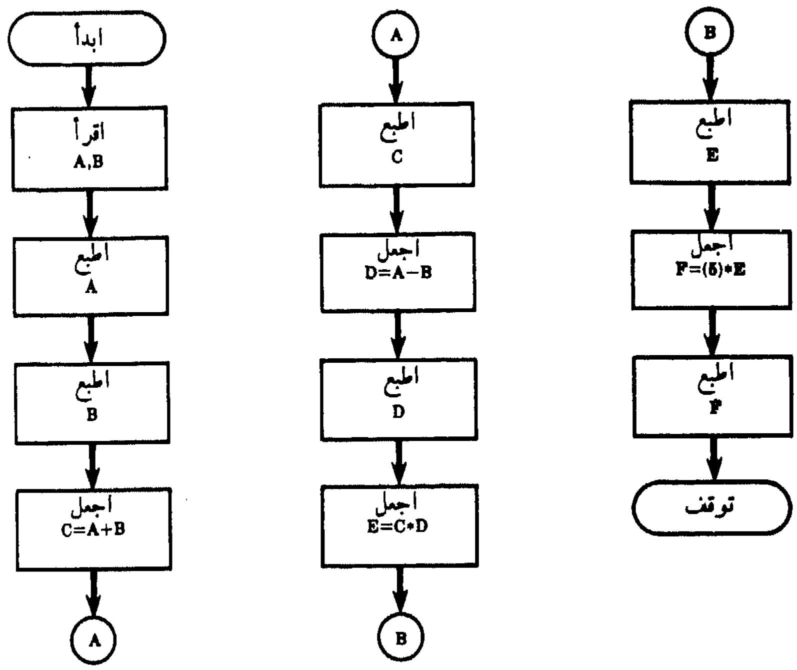
المخطط التمهيدى للبرنامج The Program Outline

دعنا نطبع عناصر كل مصفوفة فوراً بعد قراءتها أو حسابها . ويسمح لنا ذلك بملاحظة ناتج كل الخطوات فى إجراء الحسابات . وسوف تكون أيضاً نتائج كل مصفوفة ظاهرة تماماً .

وسوف نواصل الحسابات كالتالى :

١ – اقرأ عناصر المصفوفة A و B

٢ – اطبع عناصر المصفوفة A تليها عناصر المصفوفة B



شكل ٧ – ١

٣ – احسب عناصر المصفوفة C.

٤ – اطبع عناصر المصفوفة C.

٥ – احسب عناصر المصفوفة D.

٦ – اطبع عناصر المصفوفة D.

٧ – احسب عناصر المصفوفة E .

٨ – اطبع عناصر المصفوفة E .

٩ – احسب عناصر المصفوفة F .

١٠–اطبع عناصر المصفوفة F .

١١– توقف

خريطة سير العمليات المناظرة لذلك مبينة فى شكل ٧ – ١

برنامج بيسك The BASIC Program

نرى فى شكل ٧ – ٢ برنامج بيسك كامل وذلك لإجراء الحسابات . لاحظ أن البرنامج يتضمن عدة جمل مختلفة من جمل المصفوفات . لاحظ أيضاً أن البرنامج يتضمن جملة DIM ، رغم أن وجودها ليس من الأهمية حيث أن هذا البرنامج يتطلب مصفوفات 3 × 3 فقط .

```
10  REM EVALUATION OF THE MATRIX FORMULA F=(5)*(A+B)*(A-B)
20  DIM A(3,3),B(3,3),C(3,3),D(3,3),E(3,3),F(3,3)
30  MAT READ A,B
40  PRINT "THE A-MATRIX IS:"
50  MAT PRINT A;
60  PRINT "THE B-MATRIX IS:"
70  MAT PRINT B;
80  MAT C=A+B
90  PRINT "THE C-MATRIX IS:"
100 MAT PRINT C;
110 MAT D=A-B
120 PRINT "THE D-MATRIX IS:"
130 MAT PRINT D;
140 MAT E=C*D
150 PRINT "THE E-MATRIX IS:"
160 MAT PRINT E;
170 MAT F=(5)*E
180 PRINT "THE F-MATRIX IS:"
190 MAT PRINT F;
200 DATA 1,3,5,7,9,11,13,15,17,2,4,6,8,10,12,14,16,18
210 END
```

شكل ٧ – ٢

يحتوى شكل ٧ – ٢ على الخرج الذى تم توليده بواسطة هذا البرنامج ثم طباعة عناصر كل مصفوفة فى هذا البرنامج على بعد مسافات متقاربة وفى شكل جدول . وسوف نرى النتائج المطلوبة (العناصر المحسوبة للمصفوفة F) عند نهاية الشكل .

وأخيراً فإننا نوجه نظرك مرة أخرى إلى أن جمل المصفوفات ليست أساسية وكان يمكن إنجاز نفس الشئ بتضمين عدة حلقات تكرارية FOR-TO فى برنامجنا . (برنامج مكتوب بهذه الطريقة مبين فى مثال ٥ – ١٥) . إلا أن استخدام جمل المصفوفة تبسط البرمجة إلى حد بعيد .

٧ – ٣ مصفوفات خاصة SPECIAL MATRICES

وأحياناً يجب استخدام بعض المصفوفات الخاصة عند القيام بعمليات المصفوفات وذلك مثل مصفوفة **الوحدة** أو **المصفوفة المدورة** أو**مقلوب** المصفوفة . يحتوى البيسك على عدد من جمل المصفوفات التى تسمح بتكوين هذه المصفوفات الخاصة . دعنا ندرس كلا من هذه الجمل على حدة .

MAT ZER

تستخدم جملة MAT ZER لتحديد عناصر مصفوفة معينة بصفر . ويمكن أن تظهر جملة MAT ZER النموذجية كالتالى :

```
10 MAT A=ZER
```

حيث تمثل A مصفوفة لها أبعاد محددة .

```
THE A-MATRIX IS:

 1  3  5
 7  9  11
 13  15  17
THE B-MATRIX IS:

 2  4  6
 8  10  12
 14  16  18
THE C-MATRIX IS:

 3  7  11
 15  19  23
 27  31  35
THE D-MATRIX IS:

-1 -1 -1
-1 -1 -1
-1 -1 -1
THE E-MATRIX IS:

-21 -21 -21
-57 -57 -57
-93 -93 -93
THE F-MATRIX IS:

-105 -105 -105
-285 -285 -285
-465 -465 -465
```

شكل ٧ – ٣

MAT CON

والغرض من هذه الجملة هو تحديد كل عناصر مصفوفة معينة بالواحد الصحيح . ويمكن أن تكتب جملة MAT CON تماماً كالتالي :

10 MAT B=CON

حيث تمثل B مصفوفة لها أبعاد محددة .

MAT IDN

يترتب على هذه الجملة تحديد كل عناصر مجموعة متراصة مربعة بأصفار (0's) ماعدا عناصر القطر الأساسى (أى القطر الذى يتغير من أعلى اليسار إلى أسفل اليمين) حيث يتم تحديدها بالواحد الصحيح (1's) . وتعرف هذه المصفوفة التى تم تحديد عناصرها بهذه الطريقة بمصفوفة الوحدة .

ويمكن كتابة MAT IDN تماماً كالتالى :

```
10 MAT C=IDN
```

حيث تمثل المصفوفة C مصفوفة مربعة لها أبعاد محددة . (لاحظ أن المصفوفة يجب أن يكون لها نفس العدد من الأعمدة والصفوف لكى تكون مصفوفة مربعة) .

مصفوفة الوحدة لها الخصائص المهمة التالية : إذا ضربت مصفوفة مربعة D فى مصفوفة الوحدة C ، فإن حاصل الضرب سوف يكون ببساطة هو المصفوفة D بمعنى آخر :

C*D=D*C=D

(مع ملاحظة أن C و D تعمل وفق قواعد ضرب المصفوفات) . ومن ثم فإننا نرى أن ضرب المصفوفات التى تتضمن مصفوفة الوحدة مماثل للضرب العادى حيث أحد العوامل هو الثابت 1 .

مثال ٧ – ١٦

مبين فيما يلى جزء من برنامج بيسك :

```
10 DIM A(2,3),B(4,2),C(3,3)
   ...
70 MAT A=ZER
80 MAT B=CON
90 MAT C=IDN
```

سوف يسبب تنفيذ هذا البرنامج تحديد المصفوفات A و B و C بالقيم التالية :

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad B = \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{bmatrix} \qquad C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

MAT TRN

تسبب جملة MAT TRN تدوير صفوف وأعمدة مصفوفة معينة (أى تبديلها) وسوف تظهر الجملة كما يلى :

```
10 MAT B=TRN(A)
```

وبذلك فإنه إذا كانت A مصفوفة أبعادها $(m \times n)$ فإن B سوف تكون أبعادها $(n \times m)$ وتحدد عناصرها كالتالى :

B(I,J)=A(J,I)

تسمى المصفوفة B بالمصفوفة المدورة A

مثال ٧ – ١٧

ادرس برنامج البيسك التالى الذى يحتوى على الجمل التالية :

```
10 DIM A(2,3),B(3,2)
20 MAT READ A
   ...
50 MAT B=TRN(A)
   ...
80 DATA 1,3,5,7,9,11
```

الجملة رقم 20 فى تحديد عناصر المصفوفة A بالقيم الموضحة فيما بعد :

$$A = \begin{bmatrix} 1 & 3 & 5 \\ 7 & 9 & 11 \end{bmatrix}$$

وتحدد الجملة رقم 50 عناصر المصفوفة B بالقيم الموضحة فيما بعد :

$$B = \begin{bmatrix} 1 & 7 \\ 3 & 9 \\ 5 & 11 \end{bmatrix}$$

لاحظ أن المصفوفة B لها 3 صفوف وعمودين بينما المصفوفة A لها صفين و 3 أعمدة .

MAT INV

مقلوب المصفوفة المربعة هو أيضاً مصفوفة مربعة ولها الخاصية المهمة التالية : حاصل ضرب المصفوفة ومقلوبها مساو لمصفوفة الوحدة . وبمعنى آخر . فإذا كانت A مصفوفة مربعة و B مقلوب هذه المصفوفة ، من ثم :

A*B=B*A=C

حيث C هى مصفوفة الوحدة ، يجب أن تكون المصفوفة مربعة حتى يمكن حساب مقلوبها . بعض المصفوفات المربعة ، لا يمكن حساب مقلوبها . وعلى ذلك حساب مقلوب مصفوفة مربعة يكون فى بعض الحالات ممكناً وفى حالات أخرى غير ممكن .

يمكن حساب مقلوب المصفوفة (إذا كان ذلك ممكنا) بواسطة جملة MAT INV . ويمكن أن تظهر هذه الجملة كالتالى :

```
10 MAT B=INV(A)
```

حيث A و B مصفوفات مربعة لها أبعاد محددة .

يمكن الحصول على محدد المصفوفة الأصلية فور حساب مقلوب المصفوفة بواسطة الدالة المكتبية DET وترجع هذه الدالة بقيمة عددية ولا تتطلب أى خلاصة .

من بين أشياء كثيرة أخرى ، يمكن استخدام دالة DET لتحديد ما إذا كانت المصفوفة لها مقلوب أم لا ، حيث تكون قيمة المحدد صفراً فى حالة عدم إمكانية الحصول على مقلوب المصفوفة . ومع ذلك ، تذكر أنه يمكن الرجوع إلى الدالة DET فقط بعد جملة MAT INV إذا أعطيت دالة DET القيمة صفراً لمصفوفة معينة فإن مقلوب المصفوفة التى تم الحصول عليها من جملة MAT INV

السابقة ليس لها أى معنى . وعلاوة على ذلك يمكن أن يكون حساب كل من المقلوب والمحدد غير صحيحة تحت شروط معينة ، على سبيل المثال ، إذا كانت المصفوفة الرئيسية كبيرة (عدد كبير من الصفوف وعدد كبير من الأعمدة) ، أو إذا كانت مفردة تقريباً (أى أن القيمة الحقيقية لمحددها قريبة جداً من الصفر) .

مثال ٧ – ١٨ مقلوب المصفوفة Matrix Inversion

فيما يلى برنامج بيبسك بسيط يحسب مقلوب مصفوفة ، ومحدد المصفوفة وحاصل ضرب المصفوفة فى مقلوبها .

```
10 DIM A(3,3),B(3,3),C(3,3)
20 MAT READ A
30 MAT B=INV(A)
40 MAT C=A*B
50 MAT PRINT A,B,C
60 PRINT "DETERMINANT=";DET
70 DATA 5,3,1,3,7,4,1,4,9
80 END
```

وسوف ينتج عند تنفيذ هذا البرنامج ادخال القيم التالية لعناصر المصفوفات A و B و C

$$A = \begin{bmatrix} 5 & 3 & 1 \\ 3 & 7 & 4 \\ 1 & 4 & 9 \end{bmatrix} \quad B = \begin{bmatrix} 0.274854 & -0.134503 & 0.0292398 \\ -0.134503 & 0.25731 & -0.0994152 \\ 0.0292398 & -0.0994152 & 0.152047 \end{bmatrix} \quad C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

وأيضاً يكون محدد المصفوفة A مساويا 171

مبين فيما يلى الخرج الناتج من هذا البرنامج :

```
 5             3             1
 3             7             4
 1             4             9

 0.274854     -0.134503      2.92398E-2
-0.134503      0.25731      -9.94152E-2
 2.92398E-2   -9.94152E-2    0.152047

 1.            8.38190E-9    3.72529E-9
 5.58794E-9    1.            7.45058E-9
 0             7.45058E-9    1.

DETERMINANT= 171
```

لاحظ أن بعض عناصر المصفوفة C صحيح بالتقريب فقط ، وذلك نتيجة الأخطاء التى تحدث عند حسابات عناصر C . وتحدث هذه الأخطاء حين حساب الفرق بين الأرقام المتقاربة جداً فى القيمة .

حل المعادلة الآنية Solution of Simultaneous Equations

تسمح لنا جملة MAT INV لحل نظام من المعادلات الجبرية الخطية الآنية ببساطة شديدة . ولتفهم كيفية عمل الطريقة فسوف ننتفع بخصائص كل من مقلوب المصفوفة ومصفوفة الوحدة . ولنفرض مثلا أننا أعطينا مجموعة n من المعادلات :

$$\begin{aligned} c_{11}x_1 + c_{12}x_2 + \cdots + c_{1n}x_n &= d_1 \\ c_{21}x_1 + c_{22}x_2 + \cdots + c_{2n}x_n &= d_2 \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ c_{n1}x_1 + c_{n2}x_2 + \cdots + c_{nn}x_n &= d_n \end{aligned}$$

حيث تمثل c's و d's قيما معروفة و x's هى كيات غير معروفة .

فيمكن كتابة المعادلات المعطاة على هيئة مصفوفة كما يلى :

C*X=D

حيث تحتوى المصفوفة C على قيم المعاملات أى :

$$C = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1n} \\ c_{21} & c_{22} & \cdots & c_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ c_{n1} & c_{n2} & \cdots & c_{nn} \end{bmatrix}$$

وأن المتجه D به قيم الطرف الأيمن ، أى :

$$D = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_n \end{bmatrix}$$

وأن X متجه يحتوى على كيات غير معلومة :

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

دعنا نضرب معادلة المصفوفة المعطاة فى E حيث تمثل E مقلوب C فنحصل على :

E*C*X=E*D

بينما يكون حاصل ضرب المصفوفتين E*C هى مصفوفة الوحدة ، والتى سوف نشير إليها بالرمز I .. ومن ثم فيمكن كتابة :

I*X=E*D

وحيث أن I*X = X فيمكن تبسيط معادلات المصفوفات إلى :

X=E*D

وهى النتيجة المطلوبة .

والأهمية من هذه النتيجة هو التالى . إن حل نظام معادلات جبريه – خطية آنية مساو لحاصل ضرب مقلوب معاملات المصفوفة ومتجه الطرف الأيمن . يوضح مثال ٧ – ١٩ كيف يمكن تحويل هذه الفكرة ببساطة داخل برنامج بيسك .

مثال ٧ – ١٩ المعادلات الآنية Simultaneous Equations

نفرض أننا أعطينا النظام التالى المكون من 5 معادلات فى 5 مجاهيل :

$$
\begin{aligned}
11x_1 + 3x_2 \quad\quad + x_4 + 2x_5 &= 51 \\
4x_2 + 2x_3 \quad\quad + x_5 &= 15 \\
3x_1 + 2x_2 + 7x_3 + x_4 \quad\quad &= 15 \\
4x_1 \quad\quad + 4x_3 + 10x_4 + x_5 &= 20 \\
2x_1 + 5x_2 + x_3 + 3x_4 + 13x_5 &= 92
\end{aligned}
$$

ونرغب فى تحديد قيم المجاهيل x_1 و x_2 و x_3 و x_4 و x_5

طريقة إجراء الحسابات Computational Procedure

دعنا نعد كتابة المعادلات على صورة مصفوفة كالتالى :

C*X=D

حيث

$$
C = \begin{bmatrix} 11 & 3 & 0 & 1 & 2 \\ 0 & 4 & 2 & 0 & 1 \\ 3 & 2 & 7 & 1 & 0 \\ 4 & 0 & 4 & 10 & 1 \\ 2 & 5 & 1 & 3 & 13 \end{bmatrix} \qquad D = \begin{bmatrix} 51 \\ 15 \\ 15 \\ 20 \\ 92 \end{bmatrix}
$$

وسوف تحتوى X القيم x_1 و x_2 و x_3 و x_4 و x_5 . ويمكن أن نحصل على هذه القيم ببساطة بواسطة حساب حاصل ضرب المصفوفة :

X=E*D

حيث E هى مقلوب المصفوفة C .

وفور تحديد قيم x_1 إلى x_5 يمكن اختبار مدى دقة الحل بحساب حاصل الضرب :

F=C*X

فإذا تم تحديد قيم X بصورة صحيحة ، فإن المتجه F سوف يكافئ المتجه المحدد D تماماً وأى خلاف فى القيمة بين عناصر F والعناصر المناظرة فى D يعطينا مقياس الأخطاء المتضمن فى حساب X :

المخطط التمهيدى للبرنامج The Program Outline

يمكن القيام بالحساب كما يلى :

١ – اقرأ عناصر المصفوفة C والمصفوفة D .

٢ – اطبع عناصر المصفوفة C والمصفوفة D .

٣ – احسب عناصر E (وهى مقلوب المصفوفة C) .

٤ – حدد قيم X وذلك بتكوين حاصل الضرب E*D

٥ – اطبع عناصر X .

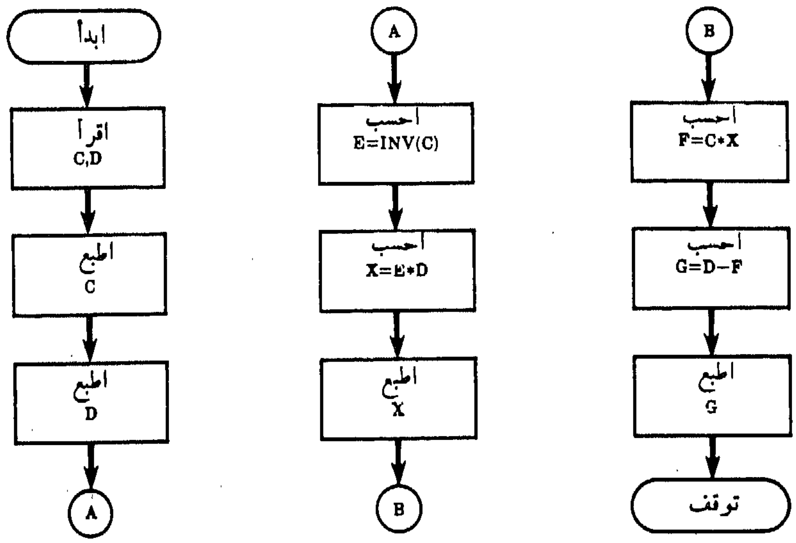
٦ – حدد قيم F وذلك بتكوين حاصل الضرب C*X

٧ – احسب متجه الأخطاء G = D — F

٨ – اطبع عناصر G .

٩ – توقف .

خريطة سير العمليات المناظرة لذلك مبينة فى شكل ٧ – ٤ .



شكل ٧ – ٤

برنامج البيسك The BASIC Program

يبين شكل ٧ – ٥ برنامج بيسك . وهيكل البرنامج مباشر تماماً ، حيث لا يتطلب هذا البرنامج التفرعات ولا الحلقات التكرارية . وهذه الطريقة المبسطة ناجمة عن استعمال جمل المصفوفات ، والتى تحرز المبرمج من تناول تفاصيل المنطق عند التعامل مع المصفوفات .

```
10  REM SOLUTION OF SIMULTANEOUS LINEAR ALGEBRAIC EQUATIONS
20  DIM C(5,5),D(5),E(5,5),F(5),G(5),X(5)
30  MAT READ C,D
40  PRINT "COEFFICIENT MATRIX:"
50  MAT PRINT C
60  PRINT "RIGHT HAND SIDE:"
70  MAT PRINT D,
80  MAT E=INV(C)
90  MAT X=E*D
100 PRINT "SOLUTION VECTOR:"
110 MAT PRINT X,
120 MAT F=C*X
130 MAT G=D-F
140 PRINT "ERROR VECTOR:"
150 MAT PRINT G,
160 DATA 11,3,0,1,2,0,4,2,0,1,3,2,7,1,0
170 DATA 4,0,4,10,1,2,5,1,3,13,51,15,15,20,92
180 END
```

شكل ٧ – ٥

نرى فى شكل ٧ – ٦ الخرج الناتج من البرنامج ، وهو يمدنا بحل 5 معادلات فى 5 مجاهيل . والحل المطلوب بالتقريب هو $x_1 = 3.0$ ، $x_2 = 2.2$ ، $x_3 = 0.21$ ، $x_4 = 0.15$ ، $x_5 = 5.7$. لاحظ أيضاً أن عناصر متجه الأخطاء G كلها أصغر فى القيمة من 10^{-6} وهذا يؤكد لنا أن الحل صحيح ومعقول .

٧ – ٤ تغيير ابعاد المجموعات المتراصة CHANGING DIMENSIONS

لقد رأينا أن جملة MAT INPUT تسمح لنا بإدخال عدد غير محدد لعناصر متجه من النهاية للطرفية المركزية ، وبذلك يمدنا بخاصية البعد المتغير . وبالإضافة إلى ذلك ، فإن عديداً من جمل المصفوفات تسمح لنا بتغيير أبعاد المجموعة المتراصة أثناء تنفيذ البرنامج ، فى الحقيقة أننا قادرون على ذلك ، إذا تطلب الأمر ، وهو تغيير حجم المجموعة المتراصة فى أماكن عديدة – بداخل البرنامج . ويمكن استخدام هذه الإمكانية لتعميم العديد من برامج البيسك التى تتضمن متجهات ومصفوفات .

مصفوفة المعاملات

11	3	0	1	2
0	4	2	0	1
3	2	7	1	0
4	0	4	10	1
2	5	1	3	13

الطرف الأيمن

51	15	15	20	92

متجه الحل

2.97917	2.2156	0.211284	0.152317	5.71503

متجه الخطأ

-9.53674E-7	-2.38419E-7	-4.76837E-7	-7.15256E-7	-1.90735E-6

شكل ٧ – ٦

جمل المصفوفات التى تسمح بإعادة تحديد حجم المصفوفة هى : MAT IDN و MAT CON و MAT ZER و MAT READ (لاحظ أن جملة MAT PRINT غير متضمنة) . تنفذ هذه الخاصية بإعطاء الحجم الحقيقى محصور بين قوسين بعد اسم المجموعة المتراصة . وفى حالة المصفوفة يجب أن نفصل بين البعدين بواسطة فاصلة (,) .

مثال ٧ – ٢٠

يحتوى برنامج بيسك على الجمل التالية :

```
10 DIM A(24,24),B(24,24)
   ...
50 MAT A=IDN(20,20)
60 MAT B=CON(8,12)
```

عند تنفيذ هذا البرنامج ، سوف يتم حجز عدد 625 عنصراً لكل من المصفوفتين A و B (متضمنة الصف رقم صفر والعمود رقم صفر) . بينما ، الجملة رقم 50 سوف تحدد A على أنها مصفوفة الوحدة وحجمها (20 × 20) وتعطى الجملة رقم 60 عناصر المصفوفة B كلها القيمة 1 وحجم B هو (12 × 8) .

يمكن أن نستخدم إما رقم ثابت أو رقم متغير في تمثيل الأبعاد الحقيقية للمجموعة المتراصة . بينما يجب أن تكون الأبعاد قيما صحيحة موجبة ولا يمكن أن تتعدى الرقم الأقصى لأحجام المجموعة المتراصة فى جملة DIM . (أما إذا كانت جملة DIM غير موجودة فإن البعد الحقيقى لا يمكن أن يتعدى الرقم 10) .

مثال ٧ – ٢١

مبين فيما يلى جزء من برنامج بيسك

```
 10 DIM  P(35),Q(50,50),R(50,50)
    ...
 50 INPUT M,N
    ...
 80 MAT READ P(M),Q(N,N)
 90 MAT R=INV(Q)
    ...
200 DATA...
```

سوف تتسبب الجملة رقم 50 فى إدخال قيم M و N (الحجم الحقيقى للمجموعة المتراصة) من النهاية الطرفية المركزية . يمكن إدخال أى أرقام صحيحة موجبة ، آخذين فى الاعتبار ألا تتعدى القيم 35 و 50 على الترتيب .

حين مقابلة الجملة رقم 80 فإن عدد M الأولى من القيم الموجودة فى كتلة البيانات سوف تعطى لعناصر المتجه P ، ثم بعد ذلك عدد (N×N) من القيم التالية (أى N↑2) سوف تعطى لعناصر المصفوفة Q. فى الجملة رقم 90 سوف يعاد تحديد حجم المصفوفة R ضمنياً وذلك بجعلها (N×N) حيث أن مقلوب المصفوفة يجب أن يكون له نفس حجم المصفوفة المعطاة .

وأخيراً ، يجب أن تكون الجملة رقم 200 موجودة وذلك لتعريف كتلة البيانات (ويمكن أن تكون عدة جمل DATA حقيقة) . لاحظ أن كتلة البيانات يجب أن تحتوى على الأقل قيم بيانات عددها M + N↑2 (وسوف تهمل أى بيانات زائدة) . أيضاً ، لاحظ أن قيما معينة من كتلة البيانات يمكن أن تعطى لعناصر المتجه P ،أو عناصر المصفوفة Q أولا تقرأ بتاتاً وذلك يتوقف على القيم المعطاة للمتغيرين M و N .

سوف نرى فى المثال التالى أن تعميم برنامج البيسك يمكن أن يحسن وذلك باستخدام خاصيه البعد المتغير .

مثال ٧ – ٢٢ توفيق المنحنى بالمربعات الصغرى (لعبة سوق الأوراق المالية)

Least Squares Curve Fitting (Playing the Stock Market)

فى هذا المثال سوف نكتب برنامج بيسك لتوافق منحنى لمجموعة من البيانات باستخدام طريقة المربعات الصغرى . وسوف نطبق هذا البرنامج بعد ذلك على مسألة توفيق « منحنى الاتجاه » المناسب لمجموعة بيانات الأرباح المبينة فى جدول ٧ – ١ لشركة وهمية (تعرف باسم Federated Mousetraps, Incorporated) وبمجرد حصولنا على « منحنى الاتجاه » سوف يكون من الممكن تحديد الأرباح لكل سهم فى وقت لاحق مثلا سنة 1978 . وقياساً على القيمة المتوسطة للنسبة بين السعر والربح ، يمكن تقدير سعر الأسهم فى سنة 1978 ، باستخدام الصيغة الرياضية :

$$P = R \cdot E$$

حيث $P =$ السعر التقديرى لسهم واحد من الأوراق المالية فى سنة 1978

$R =$ القيمة المتوسطة للنسبة بين السعر والربح .

$E =$ الأرباح التقديرية ، دولار لكل سهم ، فى سنة 1978

وسوف تستخدم هذه المعلومات كمؤشر لتقرير هل يمكن شراء أسهم الشركة بالسعر الحالى أم لا .

جدول ٧ – ١ الأرباح السنوية لكل سهم لشركة
Federated Mousetraps, Inc.

الأرباح ، دولار للسهم	السنة
$0.01	1957
0.02	1958
0.02	1959
0.03	1960
0.03	1961
0.04	1962
0.04	1963
0.09	1964
0.24	1965
0.38	1966
0.63	196[illegible]
0.93	1968
1.24	1969
1.48	1970
1.73	1971
2.07	1972
2.50	1973
3.12	1974
3.48	1975

طريقة المربعات الصغرى The Least Squares Technique

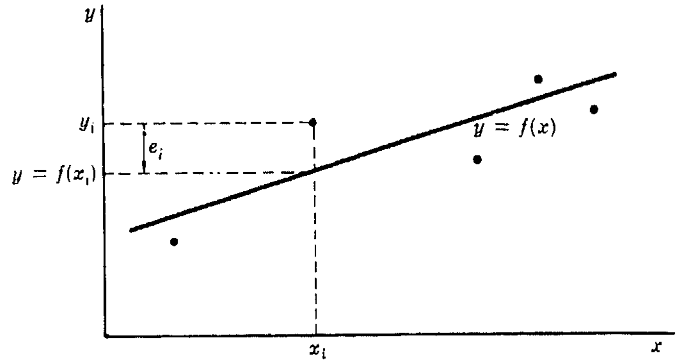
طريقة المربعات الصغرى هى طريقة شائعة لتوفيق منحنى :

$$y = f(x)$$

لمجموعة من نقط البيانات (y_1, x_1) و (y_2, x_2) و ..... و (y_M, x_M) . والطريقة مبنية على مفهوم تقليل مجموع مربعات الأخطاء :

$$e_1^2 + e_2^2 + \cdots + e_M^2$$

حيث e_i هو الخطأ رقم i وهو، لأى قيمتين معطاتين e_i و x_i ؛ الفرق بين نقطة البيانات y_i وقيمة $y = f(x_i)$ والتى تقرأ من المنحنى (انظر شكل ٧ – ٧) .



شكل ٧ – ٧

والطريقة شائعة التطبيق على الدوال الأسية والدوال المرفوعة لقوة معينة ولكثيرة الحدود . وفى كل حالة ، تتطلب الطريقة حل مجموعة من المعادلات الجبرية الخطية الآنية ، حيث الكيات المجهولة هى الكميات الثابتة لمعادلة المنحنى .

نفرض أننا نرغب فى أن تمرر الدالة ذات القوة .

$$y = ax^b$$

خلال مجموعة من نقط البيانات M ولعمل ذلك يجب أن نحل المعادلتين التاليتين لكل من a و b

$$M \log a + \left\{\sum_{i=1}^{M} \log x_i\right\} b = \sum_{i=1}^{M} \log y_i$$

$$\left\{\sum_{i=1}^{M} \log x_i\right\} \log a + \left\{\sum_{i=1}^{M} (\log x_i)^2\right\} b = \sum_{i=1}^{M} (\log x_i)(\log y_i)$$

حيث Σ تشير إلى مجموع (فمثلا $\sum_{i=1}^{M} \log x_i = \log x_1 + \log x_2 + \cdots + \log x_M$.) . لاحظ أن هذه المعادلات جبرية وخطية بدلالة المجهولين $\log a$ و b ، وبمجرد حساب قيمة $\log a$ يكون من السهل الحصول على قيمة الثابت a .

والآن نفرض أننا نرغب فى تمرير المنحنى الأسى :

$$y = ae^{bx}$$

خلال مجموعة M من نقط البيانات . والمعادلات المطلوب حلها هى :

$$M \log a + \left\{\sum_{i=1}^{M} x_i\right\} b = \sum_{i=1}^{M} \log y_i$$

$$\left\{\sum_{i=1}^{M} x_i\right\} \log a + \left\{\sum_{i=1}^{M} x_i^2\right\} b = \sum_{i=1}^{M} x_i \log y_i$$

ومرة أخرى فإننا بصدد حل معادلتين جبريتين خطيتين آنيتين لكل من $\log a$ و b .

وإذا كان المنحنى هو كثيرة الحدود :

$$y = c_1 + c_2x + c_3x^2 + \cdots + c_{n+1}x^n$$

فإننا نحدد المعاملات c_1 و c_2 و c_3 , . . . و c_{n+1} وذلك بحل المعادلات .

$$Mc_1 + \left\{\sum_{i=1}^{M} x_i\right\} c_2 + \left\{\sum_{i=1}^{M} x_i^2\right\} c_3 + \cdots + \left\{\sum_{i=1}^{M} x_i^n\right\} c_{n+1} = \sum_{i=1}^{M} y_i$$

$$\left\{\sum_{i=1}^{M} x_i\right\} c_1 + \left\{\sum_{i=1}^{M} x_i^2\right\} c_2 + \left\{\sum_{i=1}^{M} x_i^3\right\} c_3 + \cdots + \left\{\sum_{i=1}^{M} x_i^{n+1}\right\} c_{n+1} = \sum_{i=1}^{M} x_i y_i$$

. . . .

$$\left\{\sum_{i=1}^{M} x_i^n\right\} c_1 + \left\{\sum_{i=1}^{M} x_i^{n+1}\right\} c_2 + \left\{\sum_{i=1}^{M} x_i^{n+2}\right\} c_3 + \cdots + \left\{\sum_{i=1}^{M} x_i^{2n}\right\} c_{n+1} = \sum_{i=1}^{M} x_i^n y_i$$

وحيث أن الحالتين السابقتين تتضمنان حل معادلات جبرية خطية آنية فإننا سوف ندخل عدة جمل مصفوفات فى برنامجنا ، بطريقة مشابهة لما سبق ذكره فى المثال ٧ – ١٩ .

تصميم البرنامج Design of the Program

عند تصميم البرنامج سوف ندخل عدداً من الخصائص التى سوف تجعل البرنامج عام وبصورة أوسع وأشمل ، فمثلا سوف ندخل معلومات لتوافق إما دالة مرفوعة لقوة أو المنحنى الأسى التى تمت مناقشته فيما سبق ، أو كثيرة حدود حتى الدرجة التاسعة (أى لها 10 حدود) . وسوف نسمح بكية تصل إلى 100 زوج من البيانات ، وسوف نحول قيم x_i ، y_i إلى $\log x_i$ و $\log y_i$ داخلياً إذا احتجنا لهذه اللوغاريتمات . وحيث أن عدد نقط البيانات التى يجب إدخالها ونوع المنحنى المطلوب توفيقه سوف يتغير من مسألة لأخرى فإننا سوف نستخدم خاصية البعد المتغير عند كتابة البرنامج .

سوف تتم طباعة معاملات المعادلات الخطية كجزء من المخرجات . ويمكن أن تكون هذه المعلومات لها أهمية عند تتبع البرنامج لاكتشاف الأخطاء الموجودة به . كذلك فإن معادلة المنحنى المطلوب توفيقه تكون متضمنة أيضاً فى المخرجات مع عرض القيم العددية التى تم الحصول عليها لعديد من الكميات الثابتة ، ثم قائمة ببيانات الإدخال x_i و y_i مع القيمة المناظرة لها والتى تم حسابها $y(x_i)$ (وسوف يبسط ذلك عملية رسم بيانات الإدخال والمنحنى الذى تم توفيقه) ، والقيمة العددية لمجموع مربعات الأخطاء . هذه الكمية الأخيرة تفيد فى مقارنة نجاح عملية توفيق عدة منحنيات مختلفة لنفس مجموعة البيانات (وكلما كان المجموع قليلا ، كلما كان التوفيق أصلح) .

المخطط التمهيدى للبرامج The Program Outline

من أجل تخطيط البرنامج دعنا نعرف المتغيرات والمجموعات المتراصة التالية :

X = متجه له 100 عنصر يحتوى على القيم المدخلة x_i

Y = متجه له 100 عنصر يحتوى على القيم المدخلة y_i

A = مصفوفة 10 × 10 تحتوى على معاملات الكميات الثابتة المجهولة فى المعادلات الخطية .

B = مقلوب المصفوفة A

C = متجه له 10 عناصر يحتوى على الكميات الثابتة المجهولة فى المعادلات الخطية

D = متجه له 10 عناصر يحتوى على حدود الطرف الأيمن للمعادلات الخطية .

M = الكمية المدخلة التى تشير إلى عدد أزواج البيانات .

N = الكمية المدخلة التى تشير إلى المنحنى الواجب استخدامه .

$N = 0$ يشير إلى الدالة المرفوعة إلى قوة $y = ax^b$

$N = 1$ يشير إلى الدالة الأسية $y = ae^{bx}$

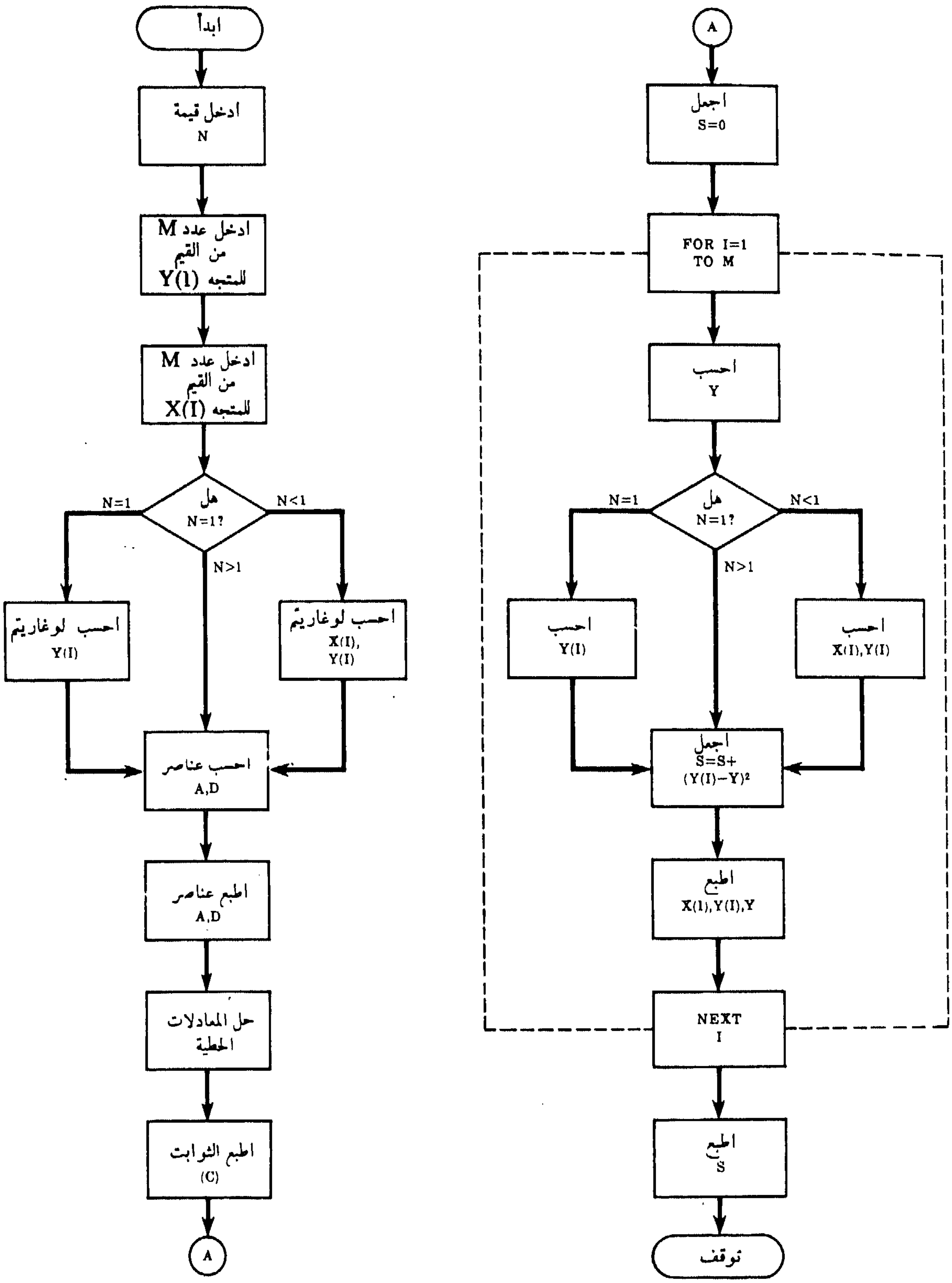
$N = 2, 3, ...10$ تشير إلى كثيرة الحدود . $y = \sum_{i=1}^{N} c_i x^{i-1}$

(فمثلا $N = 3$ تشير إلى كثيرة الحدود من الدرجة الثانية : $y = c_1 + c_2 x + c_3 x^2$)

N1 = عدد المعادلات الجبرية الخطية الآنية :

$N1 = 2$ إذا كانت $N = 0$ أو $N = 1$

$N1 = N$ إذا كانت $N = 2, 3, \ldots\ldots, 10$



ابدأ
ادخل قيمة N
ادخل عدد M من القيم للمتجه Y(I)
ادخل عدد M من القيم للمتجه X(I)
هل N=1؟
N=1
N<1
N>1
احسب لوغاريتم Y(I)
احسب لوغاريتم X(I), Y(I)
احسب عناصر A,D
اطبع عناصر A,D
حل المعادلات الخطية
اطبع الثوابت (C)
A
A
اجعل S=0
FOR I=1 TO M
احسب Y
هل N=1؟
N=1
N<1
N>1
احسب Y(I)
احسب X(I),Y(I)
اجعل S=S+ (Y(I)−Y)²
اطبع X(I),Y(I),Y
NEXT I
اطبع S
توقف

شكل ٧ – ٨

نواصل المخطط التمهيدى للبرنامج الأساسى كما يلى :

١ – نقرأ بيانات الإدخال .

(ا) نقرأ قيمة N ، وبذلك نحدد نوعية المنحنى المطلوب توفيقه .

(ب) نقرأ عدد M من القيم من أجل y_i يتبعها عدد M من القيم من أجل x_i .

٢ – احسب لوغاريتم x_i ولوغاريتم y_i إذا كانت N=0 أو احسب لوغاريتم y_i إذا كانت N=1 وذلك لكل قيم $i=1, 2, \ldots, M$

٣ – احسب عناصر المصفوفتين A و D ، باستخدام الصيغ الرياضية الملائمة للمنحنى الذى تم اختياره .

٤ – اطبع عناصر المصفوفتين A و D .

٥ – حل المعادلات الجبرية الخطية الآنية .

٦ – اطبع القيم التى تم الحصول عليها للثوابت المجهولة (عناصر المصفوفة C) فى صيغة تبين المعادلة للمنحنى المطلوب .

٧ – احسب $y(x_i)$ لكل قيم $i = 1, 2, \ldots, M$

٨ – احسب مجموع مربعات الأخطاء :

$$e_1^2 + e_2^2 + \cdots + e_M^2$$

٩ – اطبع قيم x_i و y_i و $y(x_i)$ لكل قيم $i = 1, 2, \ldots, M$. ثم بعد ذلك اطبع مجموع مربعات الأخطاء (لاحظ أنه إذا كانت N = 0 أو N = 1 فسوف يصبح من المهم تحويل لوغاريتم y_i وربما لوغاريتم x_i مرة أخرى إلى قيم y_i و x_i .

١٠ – توقف .

يبين شكل ٧ – ٨ خريطة سير العمليات لإجراء الحسابات .

برنامج البيسك **The BASIC Program**

يبين شكل ٧ – ٩ برنامج بيسك كامل . هذا البرنامج طويل إلى حد ما ويجب أن يفحص بشىء من العناية . وقد تضمن البرنامج عدداً من جمل REM وذلك من أجل تقسيم البرنامج إلى عدة كتل أساسية منطقية .

لاحظ أن البرنامج قد تضمن عدةجمل مصفوفات وكذلك دالة NUM(انظر السطور70و 75 و 90 و95و200 و 205 و370 و375).

تسمح جمل MAT INPUT بإدخال عدد متغير من نقط البيانات ، وجمل MAT ZER تسبب إعادة تحديد حجم المصفوفة A والمتجه D فى كل مرة ينفذ فيها البرنامج . وفى جمل 370 و 375 يعاد تحديد حجم المصفوفة B والمتجه C ضمنياً وذلك حتى تناظر أحجام المصفوفة A والمتجه D على الترتيب .

```
5   REM        LEAST SQUARES CURVE FITTING
10  DIM X(100),Y(100)
15  PRINT "INPUT N=0 FOR A POWER FUNCTION, N=1 FOR AN EXPONENTIAL ";
20  PRINT "FUNCTION."
25  PRINT "FOR A POLYNOMIAL, LET N EQUAL THE NUMBER OF TERMS IN ";
30  PRINT "THE POLYNOMIAL."
35  PRINT "N=";
40  INPUT N
45
50  REM        ENTER DATA POINTS
55
60  PRINT
65  PRINT "ENTER THE Y-VALUES"
70  MAT INPUT Y
75  LET M=NUM
80  PRINT
85  PRINT "ENTER THE X-VALUES"
90  MAT INPUT X
95  IF NUM=M THEN 115
100 PRINT "THE NUMBER OF Y-VALUES DOES NOT CORRESPOND TO THE ";
105 PRINT "NUMBER OF X-VALUES"
110 STOP
115
120 REM        CALCULATE LOGARITHMS OF X- AND Y-VALUES IF NECESSARY
125
130 IF N>=2 THEN 170
135 FOR I=1 TO M
140    LET Y(I)=LOG(Y(I))
145 NEXT I
150 IF N=1 THEN 170
155 FOR I=1 TO M
160    LET X(I)=LOG(X(I))
165 NEXT I
170
175 REM        CALCULATE ELEMENTS OF A-MATRIX AND D-VECTOR
180
185 LET N1=N
190 IF N1>=2 THEN 200
195 LET N1=2
200 MAT A=ZER(N1,N1)
205 MAT D=ZER(N1)
210 FOR I=1 TO N1
215    FOR J=1 TO N1
220       IF I+J>2 THEN 235
225       LET A(I,J)=M
230       GOTO 250
235       FOR K=1 TO M
240          LET A(I,J)=A(I,J)+X(K)^(I+J-2)
245       NEXT K
250    NEXT J
255    FOR K=1 TO M
260       IF I>1 THEN 275
265       LET D(I)=D(I)+Y(K)
270       GOTO 280
275       LET D(I)=D(I)+Y(K)*X(K)^(I-1)
280    NEXT K
285 NEXT I
290
295 REM        PRINT SIMULTANEOUS LINEAR EQUATIONS
300
305 PRINT
310 PRINT "COEFFICIENTS IN SYSTEM OF LINEAR EQUATIONS"
315 PRINT
320 FOR I=1 TO N1
325    FOR J=1 TO N1
330       PRINT A(I,J);
335    NEXT J
340    PRINT D(I)
345    PRINT
350 NEXT I
355
360 REM        SOLVE SIMULTANEOUS LINEAR EQUATIONS
365
370 MAT B=INV(A)
375 MAT C=B*D
380
385 REM        PRINT EQUATION FOR CURVE FIT
390
395 IF N>1 THEN 430
400 LET C1=EXP(C(1))
405 IF N=1 THEN 420
410 PRINT "POWER FUNCTION:  Y=";C1;"*X^";C(2)
415 GOTO 490
420 PRINT "EXPONENTIAL FUNCTION:  Y=";C1;"*EXP(";C(2);"*X)"
425 GOTO 490
430 IF C(2)>=0 THEN 445
435 PRINT "POLYNOMIAL FUNCTION:  Y=";C(1);C(2);"X";
440 GOTO 450
445 PRINT "POLYNOMIAL FUNCTION:  Y=";C(1);"+";C(2);"*X";
450 IF N=2 THEN 485
455 FOR I=3 TO N
460    IF C(I)>=0 THEN 475
465    PRINT C(I);"*X^";I-1;
470    GOTO 480
475    PRINT "+";C(I);"*X^";I-1;
480 NEXT I
485 PRINT
490
495 REM        PRINT INPUT VALUES OF X AND Y AND CALCULATED VALUES OF Y
500
505 IF N>=2 THEN 545
510 FOR I=1 TO M
515    LET Y(I)=EXP(Y(I))
520 NEXT I
525 IF N=1 THEN 545
530 FOR I=1 TO M
535    LET X(I)=EXP(X(I))
540 NEXT I
545 PRINT
550 PRINT TAB(2);"X";TAB(16);"Y (ACTUAL)";TAB(30);"Y (CALCULATED)"
555 LET S=0
560 FOR I=1 TO M
565    IF N>=2 THEN 595
570    IF N=1 THEN 585
575    LET Y1=C1*X(I)^C(2)
580    GOTO 615
585    LET Y1=C1*EXP(C(2)*X(I))
590    GOTO 615
595    LET Y1=C(1)
600    FOR J=2 TO N
605       LET Y1=Y1+C(J)*X(I)^(J-1)
610    NEXT J
615    LET S=S+(Y(I)-Y1)^2
620    PRINT X(I),Y(I),Y1
625 NEXT I
630 PRINT
635 PRINT "SUM OF SQUARE ERRORS=";S
640 END
```

```
INPUT N=0 FOR A POWER FUNCTION, N=1 FOR AN EXPONENTIAL FUNCTION.
FOR A POLYNOMIAL, LET N EQUAL THE NUMBER OF TERMS IN THE POLYNOMIAL.
N= ?0

ENTER THE Y-VALUES
 ?.01,.02,.02,.03,.03,.04,.04,.09,.24,.38,.63,.93,1.24,1.48,1.73&
 ?2.07,2.50,3.12,3.48

ENTER THE X-VALUES
 ?7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25

COEFFICIENTS IN SYSTEM OF LINEAR EQUATIONS

 19  51.4244 -26.0334

 51.4244  141.871 -56.6213

POWER FUNCTION:  Y= 2.26350E-7 *X↑ 5.14716

  X              Y (ACTUAL)    Y (CALCULATED)
 7.             0.01           5.06560E-3
 8              2.00000E-2     1.00722E-2
 9.             2.00000E-2     1.84678E-2
 10.            3.00000E-2     3.17640E-2
 11.            3.00000E-2     5.18788E-2
 12.            4.00000E-2     8.11884E-2
 13.            4.00000E-2     0.12258
 14.            9.00000E-2     0.179506
 15.            0.24           0.256038
 16             0.38           0.356922
 17.            0.63           0.487632
 18.            0.93           0.65443
 19.            1.24           0.864418
 20.            1.48           1.1256
 21.            1.73           1.44693
 22.            2.07           1.83839
 23.            2.5            2.31103
 24.            3.12           2.87701
 25.            3.48           3.54972

SUM OF SQUARE ERRORS= 0.614177
```

شكل ٧ — ١٠ (أ)

```
INPUT N=0 FOR A POWER FUNCTION, N=1 FOR AN EXPONENTIAL FUNCTION.
FOR A POLYNOMIAL, LET N EQUAL THE NUMBER OF TERMS IN THE POLYNOMIAL.
N= ?1

ENTER THE Y-VALUES
 ?.01,.02,.02,.03,.03,.04,.04,.09,.24,.38,.63,.93,1.24,1.48,1.73&
 ?2.07,2.50,3.12,3.48

ENTER THE X-VALUES
 ?7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25

COEFFICIENTS IN SYSTEM OF LINEAR EQUATIONS

 19  304 -26.0334

 304  5434 -213.607

EXPONENTIAL FUNCTION:  Y= 8.53337E-4 *EXP( 0.356011 *X)

  X              Y (ACTUAL)    Y (CALCULATED)
 7              0.01           1.03137E-2
 8              2.00000E-2     1.47242E-2
 9              2.00000E-2     2.10205E-2
 10             3.00000E-2     3.00094E-2
 11             3.00000E-2     4.28422E-2
 12             4.00000E-2     6.11625E-2
 13             4.00000E-2     8.73170E-2
 14             9.00000E-2     0.124656
 15             0.24           0.177962
 16             0.38           0.254062
 17             0.63           0.362705
 18             0.93           0.517806
 19             1.24           0.739238
 20             1.48           1.05535
 21             1.73           1.50664
 22             2.07           2.15091
 23             2.5            3.07069
 24             3.12           4.38379
 25             3.48           6.25839

SUM OF SQUARE ERRORS= 10.395
```

شكل ٧ — ١٠ (ب)

```
INPUT N=0 FOR A POWER FUNCTION, N=1 FOR AN EXPONENTIAL FUNCTION.
FOR A POLYNOMIAL, LET N EQUAL THE NUMBER OF TERMS IN THE POLYNOMIAL.
N= ?3

ENTER THE Y-VALUES
 ?.01,.02,.02,.03,.03,.04,.04,.09,.24,.38,.63,.93,1.24,1.48,1.73&
 ?2.07,2.50,3.12,3.48

ENTER THE X-VALUES
 ?7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25

COEFFICIENTS IN SYSTEM OF LINEAR EQUATIONS

 19  304  5434  18.08

 304  5434  105184  394.84

 5434  105184  2151370  8773.92

POLYNOMIAL FUNCTION:  Y= 1.73884 -0.34583 X+ 1.65945E-2 *X↑ 2

 X            Y (ACTUAL)    Y (CALCULATED)
 7            0.01          0.131157
 8            0.02          3.42441E-2
 9            0.02         -2.94800E-2
 10           0.03         -6.00152E-2
 11           0.03         -5.73615E-2
 12           0.04         -2.15168E-2
 13           0.04          4.75128E-2
 14           0.09          0.149733
 15           0.24          0.285143
 16           0.38          0.453741
 17           0.63          0.655528
 18           0.93          0.890505
 19           1.24          1.15867
 20           1.48          1.46002
 21           1.73          1.79457
 22           2.07          2.1623
 23           2.5           2.56322
 24           3.12          2.99733
 25           3.48          3.46463

SUM OF SQUARE ERRORS= 8.91439E-2
```

شكل ٧ — ١٠ (ج)

```
INPUT N=0 FOR A POWER FUNCTION, N=1 FOR AN EXPONENTIAL FUNCTION.
FOR A POLYNOMIAL, LET N EQUAL THE NUMBER OF TERMS IN THE POLYNOMIAL.
N= ?5

ENTER THE Y-VALUES
 ?.01,.02,.02,.03,.03,.04,.04,.09,.24,.38,.63,.93,1.24,1.48,1.73&
 ?2.07,2.50,3.12,3.48

ENTER THE X-VALUES
 ?7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25

COEFFICIENTS IN SYSTEM OF LINEAR EQUATIONS

 19  304  5434  105184  2151370  18.08

 304  5434  105184  2151370  45723424  394.84

 5434  105184  2151370  45723424  9.98814E+8  8773.92

 105184  2151370  45723424  9.98814E8  2.22672E+10  197719

 2151370  45723424  9.98814E+8  2.22672E+10  5.04212E+11  4.50758E+6

POLYNOMIAL FUNCTION:  Y=-0.727783 + 0.330811 *X-4.78821E-2 *X↑ 2 +
 2.56300E-3 *X↑ 3 -3.63439E-5 *X↑ 4

 X            Y (ACTUAL)    Y (CALCULATED)
 7            0.01          3.35158E-2
 8            0.02          1.76392E-2
 9            0.02          1.03749E-3
 10           0.03         -8.82534E-3
 11           0.03         -3.35775E-3
 12           0.04          2.21596E-2
 13           0.04          7.35737E-2
 14           0.09          0.15536
 15           0.24          0.271119
 16           0.38          0.423584
 17           0.63          0.614611
 18           0.93          0.845186
 19           1.24          1.11542
 20           1.48          1.42456
 21           1.73          1.77098
 22           2.07          2.15216
 23           2.5           2.56473
 24           3.12          3.00446
 25           3.48          3.4662

SUM OF SQUARE ERRORS= 6.42682E-2
```

شكل ٧ — ١٠ (د)

تم تبسيط حل المعادلات الآنية باستخدام جمل المصفوفات (انظر الجملتين 370 و 375) . ومن ناحية أخرى ، فقد يتطلب البرنامج عديداً من الحلقات التكرارية FOR-TO ، بالرغم من جمل المصفوفات المتاحة . وهناك عدة أسباب لذلك . أولا ، أن عناصر المصفوفة A تولد داخلياً (السطور من 185 إلى 285) . أيضاً ، يجب تحويل بيانات الإدخال إلى صيغة لوغاريتمية وبالعكس وذلك تحت شروط معينة ، (السطور من 135 إلى 145 ومن 155 إلى 165 ومن 510 إلى 520 ومن 530 إلى 540) . وأخيراً ، صياغة شكل النتائج المطلوب طباعتها تتطلب وجود عدة حلقات تكرارية FOR-TO (السطور من 320-350 ومن 455 إلى 488 ومن 650 إلى 625) وذلك من أجل إخراج البيانات .

تطبيق البرنامج Application of the Program

وبالرجوع إلى المسألة الخاصة بنا وهى سوق الأوراق المالية . دعنا نوفق منحنى الأرباح ضد بيانات الزمن للشركة Federated Mousetraps باستخدام دالة مرفوعة لقوة ومنحنى أسى لكثيرة الحدود من الدرجة الثانية (N = 3) وكثيرة حدود من الدرجة الرابعة (N = 5) ولكننا نبين قبل مواصلة الحسابات أنه فى حسابات توفيق منحنى يجب أن نراعى عدم تباعد قيم X عن قيم Y. نطرح الرقم 1950 من كل من قيم X وبذلك ، فإن السنة 1957 سوف تمثل بكل بساطة بالرقم 7 مثلا ، .:. وهكذا .

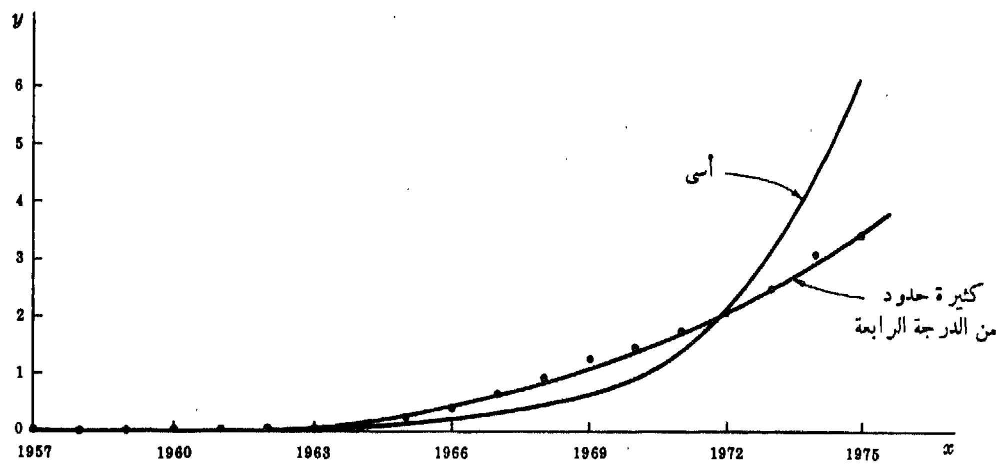
نتائج الحسابات مبينة فى الأشكال ٧ – ١٠ (ا) إلى ٧ – ١٠ (د) لكل قيم N تساوى 0 و 1 و 3 و 5 على الترتيب . وبفحص مجموع مربعات الأخطاء فى كل حالة ، نرى أن أفضل توفيق هو الذى حصلنا عليها باستخدام كثيرة الحدود من الدرجة الرابعة (N = 5) منتجة المعادلة :

$$y = -0.727783 + 0.330811\,x - 0.047882\,x^2 + 0.002563\,x^3 - 0.000036\,x^4$$

(لاحظ أن كثيرة الحدود من الدرجة الثانية أنتجت توفيقاً جيداً تقريباً) . ومن ناحية أخرى كان أسوأ توفيق هو الذى حصلنا عليها باستخدام المنحنى الأسى (N=1) ، وقد أنتج التعبير الرياضى :

$$y = 0.000853\,e^{0.356011\,x}$$

تم رسم كل من هذه المعادلات بيانياً باستخدام البيانات الأصلية فى شكل ٧ – ١١ .



شكل ٧ – ١١

من أجل تقدير الربح لكل سهم لسنة 1978 ، سوف نحسب (28 = 1950 — 1978).بالتعويض بهذه القيمة فى كثيرة الحدود من الدرجة الرابعة سوف تنتج القيمة 4.92 = y . وبذلك نرى أن الأرباح التقديرية لكل سهم من أسهم الشركة Federated Mousetraps Inc سوف تكون 4.29 $ لسنة 1978 .

وقياساً على متوسط نسبة السعر – للدخل للقيمة 30 (وهى عادية بالنسبة لشركة تنمو بهذه السرعة) ، سوف نقدر أن سهماً من أسهم الشركة (Federated Mousetraps) يجب بيعه تقريباً بمبلغ 147.60 $ فى سنة 1978 أما إذا كنا قد اشترينا المخزون فى 1974 بسعر 75.00 $ (مناظر لنسبة السعر للدخل القيمة24). فسوف يكون من المعقول أن نتوقع مضاعفة النقود تقريباً فى سنة 1978 وذلك يكافئ مكسباً حوالى 19% فى السنة بدلالة ربح بسيط يضاعف سنوياً .

كل الأمثلة المعطاة فى هذا الفصل كانت تتعلق بالمجموعات المتراصة التى تمثل عناصرها قيما عددية . ولكن نذكر القارئ ، أن البيسك يقبل أيضاً المجموعات المتراصة الحرفية وأن جمل المصفوفة يمكن أيضاً استعمالها فى القيام بعمليات المجموعات المتراصة الحرفية التقليدية . ومع كل فتذكر أن بعض نسخ البيسك لا تدعم جمل المصفوفة ، وهذا صحيح خصوصاً فى نسخ البيسك الجديدة والمتاحة للحاسبات الدقيقة .

اسئلة للمراجعة
Review Questions

٧ – ١ ما هو المتجه ؟ وما هى المصفوفة ؟

٧ – ٢ هل من الضرورى استخدام جمل المصفوفات للقيام بعمليات المتجه والمصفوفة ؟ وما هى مزايا استخدام جمل المصفوفات ؟

٧ – ٣ ما هو الفرق بين الضرب غير الموجه وضرب المصفوفة ؟ وما هى الشروط الواجب توافرها من أجل ضرب مصفوفة فى أخرى ؟

٧ – ٤ لخص قواعد كتابة كل من الجمل التالية للمصفوفات :

(ا) تحديد قيم للعناصر (ب) الجمع (ج) الطرح

(د) الضرب غير الموجه (هـ) ضرب المصفوفات

٧ – ٥ ما هو الغرض من جملة MAT READ ؟ وما هى جملة البيسك الأخرى التى يجب أن تستخدم مقترنة بهذه الجملة MAT READ

٧ – ٦ لخص قواعد كتابة جملة MAT READ .

٧ – ٧ نفرض أن جملة MAT READ تحتوى على اسم مصفوفة واحدة . فبأى ترتيب تعطى القيم الموجودة فى كتلة البيانات لعناصر المصفوفة ؟

٧ – ٨ كيف يتأثر العنصر رقم صفر للمتجه أو المصفوفة باستخدام جملة MAT READ ؟

٧ – ٩ نفرض أن جملة MAT READ واحدة تحتوى على عدة أسماء لمجموعات متراصة . فبأى ترتيب تعطى القيم الموجودة فى كتلة البيانات لعناصر المجموعات المتراصة ؟

٧ – ١٠ ما هو الغرض من جملة MAT PRINT ؟ لخص قواعد كتابة هذه الجملة .

٧ – ١١ نفرض أن جملة MAT PRINT تحتوى على اسم مصفوفة واحد . فبأى ترتيب سوف تطبع عناصر المصفوفة ؟ وكيف نميز بين صف فى مجموعة وصف آخر ، وكيف يمكن أن نغير فى التباعد بين العناصر ؟

٧ – ١٢ نفرض أن جملة MAT READ تحتوى على اسم متجه . فكيف يمكن طباعة عناصر المتجه فى شكل عمودى ؟ وفى شكل صف ؟ وكيف يمكن تغيير التباعد بين عناصر المتجه ؟

٧ – ١٣ كيف تتأثر العناصر التى تحمل أرقام صفر فى متجه أو مصفوفة بجملة MAT PRINT ؟

٧ – ١٤ نفرض أن جملة MAT PRINT واحدة تحتوى على عدة أسماء لمجموعات متراصة . فبأى ترتيب سوف تتم طباعة عناصر المجموعة المتراصة ؟ وكيف يمكن التمييز بين عناصر مجموعة متراصة وعناصر مجموعة متراصة أخرى ؟

٧ – ١٥ هل يمكن تضمين الصيغ الرياضية أو مراجع الدوال فى جملة MAT PRINT ؟

٧ – ١٦ ما هو الغرض من جملة MAT INPUT ؟ وكيف تختلف عن جملة MAT READ ؟

٧ – ١٧ لخص قواعد كتابة جملة MAT INPUT .

٧ – ١٨ ماذا يحدث عند تنفيذ جملة MAT INPUT ؟ وكيف تنقل قيم البيانات المدخلة للحاسب ؟

٧ – ١٩ كيف يتأثر العنصر رقم صفر بواسطة جملة MAT INPUT ؟

٧ – ٢٠ كيف يمكن قراءة سطر أو أكثر من بيانات الإدخال بواسطة جملة MAT INPUT واحدة ؟

٧ – ٢١ كيف يمكن تحديد عدد عناصر المتجه التي تم إدخالها بواسطة جملة MAT INPUT اقترح مكاناً ملائماً لتخزين هذا الرقم .

٧ – ٢٢ هل يمكن لجملة MAT INPUT أن تستخدم لإدخال عناصر مصفوفة كما هو الحال مع عناصر المتجه ؟ هل هناك أى قيود على المصفوفات التى لا توجد مع المتجهات ؟

٧ – ٢٣ ما هى مصفوفة الوحدة ؟

٧ – ٢٤ ما هو المقصود من تدوير مصفوفة ؟

٧ – ٢٥ ما هو المقصود من إيجاد مقلوب المصفوفة ؟ هل يمكن الحصول على مقلوب المصفوفة دائماً ؟ اشرح .

٧ – ٢٦ ما هى العلاقة التى يجب أن توجد ما بين أرقام الصفوف وأرقام الأعمدة فى مصفوفة الوحدة ؟ وفى مقلوب مصفوفة ؟ (أو مصفوفة مطلوب إيجاد مقلوبها) ؟

٧ – ٢٧ ما هى العلاقة التى يجب أن توجد ما بين أرقام السطور وأرقام الأعمدة لمصفوفة معينة والمصفوفة الناتجة من تدويرها ؟

٧ – ٢٨ ما هى نتيجة ضرب مصفوفة بمصفوفة الوحدة ؟

٧ – ٢٩ ما هى نتيجة ضرب مصفوفة بمقلوبها ؟

٧ – ٣٠ اذكر الغرض ولخص قواعد كتابة كل من جمل المصفوفات التالية :

(a) MAT ZER
(b) MAT CON
(c) MAT IDN
(d) MAT TRN
(e) MAT INV

٧ – ٣١ ما هو الغرض من الدالة المكتبية DET ؟ ولأى نوع من أنواع المصفوفات يمكن أن تستخدم دالة DET ؟ وما هى جملة المصفوفة التى يجب أن تسبق دائماً الإشارة إلى دالة DET ؟

٧ – ٣٢ كيف يمكننا أن نقرر ما إذا كانت المصفوفة المربعة يمكن الحصول على مقلوبها أم لا ؟

٧ – ٣٣ كيف يمكننا حل المعادلات الجبرية الخطية الآنية باستخدام جمل المصفوفة ؟ هل يوجد إجراء معين للتحقق من دقة الحل ؟

٧ – ٣٤ ما هى جمل المصفوفة التى تسمح بإعادة تحديد حجم مجموعة متراصة أثناء تنفيذ البرنامج ؟ وكيف يمكن إنجاز إعادة تحديد حجم المجموعة المتراصة ؟

٧ – ٣٥ قارن بين مفهوم إعادة تحديد حجم المجموعة المتراصة التى تمت مناقشتها فى القسم ٧ – ٤ وبين خاصية البعد المتغير المتضمن فى جملة MAT INPUT .

٧ – ٣٦ ما هى العمليات التى يمكن تأديتها على المجموعات المتراصة الحرفية ؟ وما هى جمل المصفوفة التى يمكن استخدامها فى مناولة المجموعات المتراصة الحرفية ؟

مسائل محلولة
Solved Problems

٧ – ٣٧ مبين فيما يلى عدة جمل أو مجموعات من جمل البيسك . بعض منها مكتوب بصورة غير صحيحة تعرف على كل الأخطاء :

(ا)

```
10 DIM P(12,20),Q(12,20),R(12,20)
   ...
50 MAT P=(.5)*(Q+R)
```

لا يمكن أن تظهر صيغ رياضية للمصفوفة فى جملة المصفوفة .

(ب)

```
10 DIM P(12,20),Q(12,20)
   ...
50 MAT P=((N+1)/2)*Q
```

صحيح ، مادامت N متغير عادى (غير متجه) .

(جـ)

```
30 MAT INPUT A,B,C
```

معظم نسخ البيسك تسمح بظهور اسم متجه واحد فقط فى جملة MAT INPUT .

(د)

```
30 MAT INPUT N$
```

صحيحة ، مادامت N$ هى متجه حرفى .

(هـ)

```
200 MAT PRINT A,B,A+B,A-B
```

لا يمكن أن تظهر الصيغ الرياضية فى جملة MAT PRINT

(و)

```
10 DIM A(10,20),B(20,10)
   ...
30 MAT READ A
40 MAT B=TRN(A)
```

صحيحه

(ز)

```
10 DIM P(3,6),Q(8,10),R(5,5)
   ...
50 MAT R=P+Q
```

لا يمكن جمع المصفوفات ما دامت أبعادها غير متناظرة .

(ح)

```
10 DIM C(6,8),D(6,8),V(12)
   ...
60 MAT D=INV(C)
   ...
90 MAT V=IDN
```

من الاستحالة حساب مقلوب مصفوفة غير مربعة . وأيضاً لا يمكن أن يظهر متجه فى جملة MAT IDN

٧ – ٣٨ اكتب جملة أو أكثر من جمل البيسك لكل من المواقف الموصوفة التالية :

(أ) احسب قيمة الصيغة الرياضية

$$Y=X^T*A*X$$

حيث A مصفوفة (10×10) و X من (10) عناصر (وذلك بإهمال المنصر رقم صفر) و X^T هي مصفوفة X المدورة . ما هي أبعاد Y ؟

```
100 MAT Z=TRN(X)
110 MAT B=Z*A
120 MAT Y=B*X
```

أو

```
100 MAT Z=TRN(X)
110 MAT C=A*X
120 MAT Y=Z*C
```

(سوف تمثل Y قيمة واحدة (أي أن Y سوف تكون متغيراً عادياً وليس مجموعة متراصة)

(ب) احسب قيمة الصيغة الرياضية :

$$T=(N)*F^{-1}*G+H$$

حيث F و G و H مصفوفات (50 × 50) و N متغير عادى (غير متجه) و F^{-1} تمثل مقلوب المصفوفة F . ما هي أبعاد T ؟

```
100 MAT A=INV(F)
110 MAT B=A*G
120 MAT C=(N)*B
130 MAT T=C+H
```

سوف تكون T مصفوفة (50 × 50)

(ج) احسب الفرق بين I و $A*A^{-1}$ حيث A مصفوفة (10 × 10) و A^{-1} مصفوفة تمثل مقلوب المصفوفة A و I هي مصفوفة الوحدة (10 × 10) (لاحظ أن هذا الفرق من حيث المبدأ يجب أن يكون مصفوفة عناصرها صفرا . وفى الحقيقة فإن عناصر المصفوفة يمكن أن تختلف عن الصفر وذلك بسبب الأخطاء الرقية) .

```
100 MAT B=INV(A)
110 MAT C=B*A
120 MAT I=IDN
130 MAT D=I-C
```

(د) اطبع الإجابة التي تم الحصول عليها فى الجزء (ج) السابق فى شكل مصفوفة ، مع تقارب عناصرها على قدر المستطاع .

```
140 MAT PRINT D;
```

(هـ) نفرض أن A و B مصفوفتان (10 × 10) حيث تعطى عناصرهما فى كتلة بيانات . اقرأ عناصر A ، صفاً فى كل مرة ، ثم تتبعها عناصر B عموداً فى كل مرة .

```
10 MAT READ A,C
20 MAT B=TRN(C)
```

(لاحظ أن A و C تقرأ صف صف)

(و) نفرض أن X و Y متجهات يجب أن تحدد قيم عناصرها بقيم تعطى من النهاية الطرفية المركزية . وسوف تخزن عدد القيم التي تم إدخالها لكل من X و Y في X(0) و Y(0) على الترتيب .

```
10 MAT INPUT X
20 LET X(0)=NUM
30 MAT INPUT Y
40 LET Y(0)=NUM
```

(ز) نفرض أن P مصفوفة (12 × 5) و T مصفوفة (6 × 6) و Y متجه له (10) عناصر . اطبع قيم P و T و X حيث تظهر قيم X في شكل صف .

```
200 MAT PRINT P,T,X,
```

(ح) كرر الجزء (ز) السابق ، مع تقارب عناصر المجموعة المتراصة على قدر المستطاع .

```
200 MAT PRINT P;T;X;
```

مسائل تكميلية
Supplementary Problems

٧ – ٣٩ مبين فيما يلي عدة جمل أو مجموعات من جمل البيسك . بعضها كتب بطريقة خاطئة . تعرف على كل الأخطاء :

(أ)

```
10 DIM L$(100),M$(100)
 . . .
50 MAT INPUT M$
60 MAT L$=M$
```

(ب)

```
35 MAT Q=(A+B)*P
```

حيث A و B و P و Q مصفوفات (10 × 10)

(جـ)

```
35 MAT Q=(A+B)*P
```

حيث A و B متغيرات عادية (غير متجه) و P و Q مصفوفات (10 × 10)

(د)

```
60 MAT H=(3)*H
```

حيث H مصفوفة 6 × 6

(هـ)

```
60 MAT G=H*H
```

حيث G و H مصفوفتان (6 × 6)

(و)

```
60 MAT H=G*H
```

حيث G و H مصفوفتان (6 × 6)

(ز)

```
10 DIM X(10,20),Y(10,20),Z(10,20)
 . . .
50 MAT Z=X*Y
```

(ح)

```
10 DIM V(100)
 . . .
50 MAT INPUT V(50)
```

(ط)

```
10 DIM C(100,50),D(100,50)
20 INPUT M,N
30 MAT READ C(M,N),D(M,N)
```

(ى)

```
10 DIM C(100,50),D(100,50)
20 INPUT M,N
...
100 MAT PRINT C(M,N),D(M,N)
```

(ك)

```
10 DIM A(25,25)
...
50 MAT A=CON(12,24)
```

(ل)

```
10 DIM A(20,5)
...
80 MAT A=ZER(10,6)
```

(م)

```
10 DIM F(10,20),G(10,20)
...
60 MAT G=INV(F)
```

(ن)

```
10 DIM K(20,30)
...
50 MAT K=IDN(12,12)
```

٧ ـ ٤ اكتب جملة أو أكثر من جمل البيسك لكل من المواقف الموصوفة فيما يلى :

(ا) احسب قيمة الصيغة الرياضية .

F=(2*N+1)*(AT*A−I)

حيث A مصفوفة (10 × 10) و A^T مصفوفة A المدورة ، I هى مصفوفة الوحدة (10 × 10) و N متغير عادى (غير متجه) . ماذا ستكون أبعاد F ؟

(ب) افرض أن A و B و C و D مصفوفات 10 × 10 احسب محدد G حيث :

G=A*C−B*D

ثم اطبع نتيجته .

(جـ) عدل جزء (ب) السابق وذلك لطباعة عناصر G يتبعها مقلوب G ثم محدد G . قارب بين عناصر المصفوفة على قدر المستطاع .

(د) نفرض أن A مصفوفة (30 × 20) حدد قيم أول 8 أعمدة من أول 12 صف بصفر .

(هـ) نفرض أن A و B مصفوفتان (30 × 20) . اقرأ العناصر فى الأعمدة الإثنى عشر الأولى من الثمانية صفوف الأولى فى المصفوفة A ، تتبعها عناصر الخمسة عشرة الأولى من الستة صفوف الأولى من المصفوفة B . كيف يجب تنظيم البيانات فى كتلة البيانات ؟

(و) نفرض أن A و B مصفوفتان (30 × 20) اطبع العناصر الموجودة فى الإثنى عشر عموداً الأولى من الثمانية صفوف الأولى من المصفوفة A ،تتبعها عناصر الأعمدة الخمسة عشر الأولى من الستة صفوف الأولى من المصفوفة ثم قارن بالحل الموجود فى الجزء (هـ) السابق .

مسائل للبرمجة
Programming Problems

٧ - ٤١ عدل البرنامج المعطى فى مثال ٧ - ١٥ حيث يمكن حساب المصفوفة المدورة للمصفوفة F ،ثم مقلوبها ثم محدد المصفوفة F واطبعها . اشرح النتائج التى سوف تحصل عليها .

٧ - ٤٢ عدل البرنامج المعطى فى مثال ٧ - ١٩ حتى يمكن حل أى عدد من N معادلة فى N مجهول . استخدم خاصية البعد المتغير الموصوف فى قسم ٧ - ٤ . ادخل وسيلة لطباعة محدد معاملات المصفوفة .

٧ - ٤٣ عند توليد أخطاء رقية فى حل المعادلات الآنية C∗X = D فمن اليسير أن تواصل كما يلى :

(ا) احسب المتجه F = C∗Y

(ب) احسب متجه الأخطاء G = D − F (إذا لم توجد أخطاء رقية فإن F ستبقى كما هى كالمتجه المعطى D ، وعناصر G سوف تكون أصفاراً) .

(ﺟ) إن لم تكن عناصر G كلها أصفار فحل المعادلات الآنية التالية :

A∗Y=G

للقيم Y(1) و Y(2) و ...., Y(N)

(د) اجمع عناصر Y على العناصر المناظرة فى X وذلك للحصول على حل معدل .

عدل البرنامج المعطى فى مثال ٧ - ١٩ متضمناً خاصية تصحيح الأخطاء .أدخل وسيلة لحساب محدد معاملات المصفوفة .

استخدم البرنامج لحل المعادلات الآتية :

$$
\begin{aligned}
x_1 + \tfrac{1}{2}x_2 + \tfrac{1}{3}x_3 + \tfrac{1}{4}x_4 + \tfrac{1}{5}x_5 &= \tfrac{1}{6}\\
\tfrac{1}{2}x_1 + \tfrac{1}{3}x_2 + \tfrac{1}{4}x_3 + \tfrac{1}{5}x_4 + \tfrac{1}{6}x_5 &= \tfrac{1}{7}\\
\tfrac{1}{3}x_1 + \tfrac{1}{4}x_2 + \tfrac{1}{5}x_3 + \tfrac{1}{6}x_4 + \tfrac{1}{7}x_5 &= \tfrac{1}{8}\\
\tfrac{1}{4}x_1 + \tfrac{1}{5}x_2 + \tfrac{1}{6}x_3 + \tfrac{1}{7}x_4 + \tfrac{1}{8}x_5 &= \tfrac{1}{9}\\
\tfrac{1}{5}x_1 + \tfrac{1}{6}x_2 + \tfrac{1}{7}x_3 + \tfrac{1}{8}x_4 + \tfrac{1}{9}x_5 &= \tfrac{1}{10}
\end{aligned}
$$

قارن الإجابات التى تم الحصول عليها بدون خاصية تصحيح الأخطاء .

٧ - ٤٤ عدل البرنامج المعطى فى مثال ٧ - ٢٢ حتى يمكن توفيق عدة منحنيات مختلفة لنفس مجموعة البيانات (أى يمكن أن يشغل البرنامج عدة مرات مختلفة) بدون إعادة قراءة البيانات كل مرة يعاد فيها البرنامج .

٧ - ٤٥ اكتب برنامج بيسك لكل مسألة موصوفة فيما يلى . ضمن الحل مخطط تفصيلى أو خريطة لسير عمليات للمسألة .

(ا) مبتدئاً بالمصفوفة A المعطاة فى مثال ٧ - ١٨ ، احسب مصفوفة D جديدة حيث كل عنصر من عناصر المصفوفة D هو مقلوب العنصر المناظر فى المصفوفة A أى $d_{ij} = 1/a_{ij}$. قارن عناصر المصفوفة D بعناصر مقلوب المصفوفة A (مقلوب المصفوفة ممثل فى المصفوفة B فى المثال ٧ - ١٨) .

(ب) احسب مقلوب المصفوفة B المعطاة فى مثال ٧ - ١٨ . قارن عناصر هذه المصفوفة بعناصر المصفوفة A فى المثال ٧ - ١٨ .

(ﺟ) وضح أن A∗B لا تساوى B∗A لكل من المصفوفتين A و B المعطاة فى المثال ٧ - ١٥ .

(د) وضح أن $(A*B)^T$ تساوى B^T*A^T لكل من المصفوفتين A و B المعطيتين فى مثال ٧ – ٦ ، حيث A^T تمثل المصفوفة A المدورة . و B^T تمثل المصفوفة B المدورة .. إلخ .

(هـ) احسب قيمة عناصر المصفوفة :

$$Y=D^T*C^T*C*D$$

باستخدام المصفوفة C والمتجه D المعطاة فى مثال ٧ – ١٩ . ما هى أبعاد Y ؟

(و) احسب قيمة عناصر المصفوفة

$$P=I+A+A^2+A^3+A^4$$

باستخدام المصفوفة A فى مثال ٧ – ١٨ (لاحظ أن I تمثل مصفوفة الوحدة و $A^2 = A*A$ و $A^3 = A*A*A$ ... إلخ) ما هى أبعاد P ؟

الفصل ٨

ملفـــات البيـــانات
Data Files

الملف فى اللغة الاصطلاحية عبارة عن مجموعة منظمة ومستقلة من المعلومات . ويمكن أن يتضمن أى نوع من المعلومات . من ثم فيمكن أن يحتوى الملف على سلسلة متعاقبة من جمل بيسك ، أو يتكون من قيم للبيانات (أى أرقام وحروف) . النوع الأول هو بالطبع برنامج بيسك ويسمى الأخير ملف بيانات .

تعتبر ملفات البيانات وسيلة مناسبة لتخزين مجموعة من البيانات حيث يمكن قراءتها وتعديلها بواسطة برنامج بيسك ويتناول هذا الفصل إنشاء واستخدام مثل هذه الملفات .

وسوف نرى أن البيسك يتضمن عدداً من الجمل للتعامل مع الملفات ، مماثلة لجمل المصفوفات التى تم تقديمها فى الفصل السابع . إلا أنها تختلف عن جمل المصفوفات فى وجود كثير من الاختلافات فى الشكل من نسخة بيسك لأخرى . ولذلك فإن هذا الفصل سوف يؤكد مفهوم التعامل مع ملف البيانات دون الدخول فى تفاصيل الجمل كل على حدة . (ظهور جمل التعامل مع الملف فى هذا الفصل جزء من لغة بيسك المطبقة على حاسب Digital Equipment Corporations أى نظام DEC-10 . وهى نموذج لجمل التعامل مع الملفات المتاحة فى نسخ أخرى من البيسك . أما إجراءات التعامل مع الملفات المستخدمة مع الحاسبات الدقيقة فيتم تقديمها فى الفصل التاسع بالقسم ٩ – ٤ .

٨ – ١ ملفات البيانات التسلسلية SEQUENTIAL DATA FILES

من أهم خصائص الملفات التسلسلية فى واقع الأمر أن عناصر البيانات تنظم كلا على حدة بطريقة تسلسلية ، أى واحدة تلو الأخرى . وعادة تناظر عناصر هذا النوع من الملفات السطور المنفصلة من المعلومات على وحدة الكونسول . وبذلك فإن برنامج بيسك سوف يخزن كلف تسلسل ، حيث كل جملة سوف تظهر على سطر منفصل وتنظم الجمل بترتيب تصاعدى لأرقام السطور .

يمكن أن تمثل الملفات التسلسلية أيضاً مجموعات من البيانات . وسوف تتكون مثل هذه الملفات من عدة سطور من البيانات ، كل سطر منها يبدأ برقم سطر . وسوف تنظم السطور فى تسلسل بترتيب تصاعدى لأرقام السطور . يمكن أن تكون عناصر البيانات فى سطر أرقاماً أو حروفاً أو توافقية من الاثنين ، يفصل بين كل منهما فاصلة (,) أو أماكن خالية إذا كانت سلسلة الحروف بها فاصلة (,) أو مكان خال ، فيجب أن تنحصر بين علامتى اقتباس .

مثال ٨ – ١

يحتوى ملف بيانات تسلسلى على الاسم ودرجات الاختبارات لكل طالب فى فصل دراسى يدرس علوم الحاسب . سوف يظهر الملف كما يلى :

```
10 "COMP SCI 100" "FALL, 1982"
20 "ADAMS B F" 45 80 80 95 55
30 "BROWN P" 60 50 70 75 55
40 "DAVIS R A" 40 30 10 45 60
50 "FISHER E K" 0 5 5 0 10
60 "HAMILTON S P" 90 85 100 95 90
70 "JONES J J" 95 90 80 95 85
80 "LUDWIG C W" 35 50 55 65 45
90 "OSBORNE T" 75 60 75 60 70
100 "PRINCE W F" 85 75 60 85 90
110 "RICHARDS E N" 50 60 50 35 65
120 "SMITH M C" 70 60 75 70 55
130 "THOMAS B A" 10 25 35 20 30
140 "WOLFE H" 25 40 65 75 85
150 "ZORBA D R" 65 80 70 100 60
```

نرى أن كل سطر يبدأ برقم للسطر . يحتوى السطر الأول على سلسلتين من الحروف ، لتعرف المادة الدراسية والفصل الدراسى على الترتيب . (لاحظ أن سلسلة الحروف تحصر بين علامتى الاقتباس ، حيث تحتوى كل من السلسلتين على الفاصلة (,) والمسافة الخالية) . يحتوى كل من السطور المتعاقبة على اسم الطالب (سلسلة من الحروف) ، تتبعها خمس درجات للامتحانات . وبذلك فإن الملف يتكون من 15 سطراً ، بالرغم من وجود 14 طالباً فقط فى الفصل .

ويجب أن يكون مفهوماً أن تتابع ملف البيانات الذى نحن بصدده يتقرر بأرقام السطور وليس بالأسماء ، ولا أهمية فى الحقيقة أن تكون الأسماء مرتبة ترتيباً هجائياً .

انشاء وتنقيح ملف متتابع Creating and Editing a Sequential File

لما كان ملف البيانات يبنى بنفس طريقة بناء برنامج بيسك فيمكننا إنشاء وتنقيح مثل هذا الملف بنفس الطريقة المتبعة من البرنامج . ويمكننا أيضاً طباعة ملف بيانات متتابع على الكونسول إذا رغبنا فى ذلك . تنفذ هذه الوظائف مع أوامر النظام الممثلة فى الفصل الثالث (مثال ذلك NEW و OLD و SAVE و LIST و .... وإلخ) .

قواعد إعطاء اسم للملف البيانات هى نفس القواعد المستخدمة عند إعطاء اسم لبرنامج بيسك . والصورة التقليدية لذلك هى من حرف إلى ستة أحرف ويجب أن تبدأ بحرف هجائى

مثال ٨ – ٢

نفرض أننا نريد إدخال ملف البيانات المتتابع المبين فى مثال ٨ – ١ للحاسب من خلال الكونسول وتخزينه تحت اسم SCORES ولتنفيذ ذلك يجب أن نسجل دخولنا للنظام ، ثم نعرف أن اسم الملف الجديد هو SCORES ، ثم نطبع مجموعة البيانات سطراً بسطر ثم بعد ذلك نخزن الملف . ويمكننا أيضاً ، إذا رغبنا فى ذلك ، عمل قائمة بالملف ، بعد إتمام إدخاله وتصحيحه .

يبين شكل ٨ – ١ الجزء الأساسى من جلسة المشاركة الزمنية التى تبدأ بعد إجراء التسجيل لدخول النظام . لاحظ حدوث ثلاثة أخطاء طباعة عند إدخال البيانات (فى السطور 120 و 130 و 150) تم تصحيح هذه الأخطاء إما من خلال استخدام مفتاح DELETE (السطور 120 و 130) أو بإعادة طبع السطر بالكامل (انظر السطر 150) .

قائمة كاملة بملف البيانات مبينة فى نهاية الشكل ٨ – ١ . تنتج هذه القائمة بناء على أمر LIST ، المبين فى منتصف الشكل . وكل أوامر النظام التى يدخلها المستخدم موضوع تحتها خط .

يمكن أيضاً إنشاء ملف البيانات المتتابع مباشرة بواسطة برنامج البيسك وسوف نرى كيف يتم تنفيذ ذلك فى مثال ٨ – ٤ .

قراءة ملف بيانات متتابع Reading a Sequential Data File

فى معظم التطبيقات سوف تقرأ المعلومات المخزنة فى ملف البيانات ثم تشغل بواسطة برنامج بيسك . ويجب أن تقرأ بنود ، وتبدأ من ملف البيانات المتتابع بنفس طريقة التخزين وبنفس الترتيب . مبتدءًا من بداية ملف البيانات . وسوف تحفظ البيانات التى تمت قراءتها لاستخدامها بعد ذلك .

ومن أجل قراءة البيانات من ملف بيانات متتابع وتحت تحكم البرنامج ، سوف نستخدم جمل التعامل مع الملفات وهى FILES و INPUT و IF END ويوضح المثال التالى كيف يتم إنجاز ذلك .

```
NEW OR OLD-->NEW
NEW FILE NAME-->SCORES
>10  "COMP SCI 100" "FALL, 1982"
>20  "ADAMS B F" 45 80 80 95 55
>30  "BROWN P" 60 50 70 75 55
>40  "DAVIS R A" 40 30 10 45 60
>50  "FISHER E K" 0 5 5 0 10
>60  "HAMILTON S P" 90 85 100 95 90
>70  "JONES J J" 95 90 80 95 85
>80  "LUDWIG C W" 35 50 55 65 45
>90  "OSBORNE T" 75 60 75 60 70
>100 "PRINCE W F" 85 75 60 85 90
>110 "RICHARDS E N" 50 60 50 35 65
>120 "SMITH M C" 70 60 7%\%\5 70 55
>130 "THON\N\MAS C\C\B A" 10 25 35 20 30
>140 "WOLFE H" 25 40 65 75 85
>150 "ZORBA 65 80 70 100 60
>150 "ZORBA D R" 65 80 70 100 60
>SAVE
>LIST

SCORES

10  "COMP SCI 100" "FALL, 1982"
20  "ADAMS B F" 45 80 80 95 55
30  "BROWN P" 60 50 70 75 55
40  "DAVIS R A" 40 30 10 45 60
50  "FISHER E K" 0 5 5 0 10
60  "HAMILTON S P" 90 85 100 95 90
70  "JONES J J" 95 90 80 95 85
80  "LUDWIG C W" 35 50 55 65 45
90  "OSBORNE T" 75 60 75 60 70
100 "PRINCE W F" 85 75 60 85 90
110 "RICHARDS E N" 50 60 50 35 65
120 "SMITH M C" 70 60 75 70 55
130 "THOMAS B A" 10 25 35 20 30
140 "WOLFE H" 25 40 65 75 85
150 "ZORBA D R" 65 80 70 100 60
```

شكل ٨ – ١

مثال ٨ – ٣

دعنا نرجع إلى قائمة الأسماء ودرجات الامتحان المبينة فى شكل ٨ – ١ . ونفرض أننا نريد القيام بالعمليات التالية لكل طالب فى الفصل .

١ – اقرأ درجات الاختبارات الخمسة الأوائل من ملف بيانات SCORES

٢ – ثم أدخل درجة اختبار سادسة من الكونسول .

٣ – احسب المتوسط لكل من الدرجات الست المعطاة .

٤ – اطبع المتوسط على الكونسول .

وسوف تتوقف الحسابات عند الوصول إلى نهاية ملف البيانات .

وسوف يقوم برنامج بيسك بالعمليات المبينة فى شكل ٨ — ٢ . وتعرف المتغيرات الموجودة فى البرنامج كالتالى :

N = رقم السطر لكل سطر من ملف البيانات SCORES

T$ = عنوان المادة الدراسية :

Y$ = الفصل الدراسى .

N$ = اسم الطالب

C1 و C2 و C3 و C4 و C5 = درجات الاختبارات لكل طالب فى ملف البيانات SCORES

C6 = درجة الاختبار التى تدخل من الكونسول لكل طالب .

A = المتوسط المحسوب للاختبارات الستة للطالب .

فيما يلى شرح لجمل التعامل مع الملف التى تظهر فى هذا البرنامج .

```
10  FILES SCORES
20  INPUT #1,N,T$,Y$
30  PRINT "COURSE:";T$,"TERM:";Y$
40  PRINT
50  INPUT #1,N,N$,C1,C2,C3,C4,C5
60  PRINT N$,"SCORE=";
70  INPUT C6
80  LET A=(C1+C2+C3+C4+C5+C6)/6
90  PRINT "AVERAGE=";A
100 PRINT
110 IF END #1,THEN 130
120 GOTO 50
130 END
```

شكل ٨ — ٢

يتم تبادل المعلومات بين ملف البيانات وبرنامج البيسك من خلال قناة البيانات . تحدد الجملة رقم 10 (FILES) ملف البيانات SCORES لقناة البيانات رقم 1 . ويجب أن يحدث ذلك قبل تبادل المعلومات من أو إلى ملف البيانات . وكل جمل التعامل مع البيانات التى تعقب ذلك سوف تشير إلى ملف البيانات برقم القناة وليس باسمه .

تقرأ الجملة رقم 20 رقم السطر N وعنوان المادة الدراسية T$ والفصل الدراسى Y$ من أول سطر من ملف البيانات الذى تحددت له قناة رقم 1 ، (أى ملف البيانات SCORES) . تقرأ الجملة 50 ،رقم سطر واسم الطالب و 5 درجات اختبار من كل سطر فى SCORES (لاحظ أن أرقام السطور يجب أن تقرأ من ملف البيانات ، بالرغم من أنها لا تستخدم) .

تختبر الجملة 110 نهاية ملف البيانات . إذا لم توجد أى بيانات أخرى (نهاية الملف) ، فإن التحكم سوف يتحول إلى الجملة 130 (END) وإلا فسوف يتحول التحكم إلى الجملة المنفذة التالية (الجملة 120) . وبذلك نرى أن هذه الجملة تشبه كثيراً جملة IF–THEN الشهيرة ، بالرغم من أنها تستخدم فقط لتختبر شرط نهاية الملف .

باقى جمل هذا البرنامج عبارة عن جمل بيسك عادية ، تماماً كالتى نوقشت فى فصول سابقة من هذا الكتاب ، ومعانيها يجب أن تكون واضحة .

يبين شكل ٨ — ٣ الخرج الناتج من هذا البرنامج وملف البيانات SCORES. (قائمة بملف البيانات مبين فى شكل ٨ — ١) . لاحظ أن المعلومات التى تم إدخالها مباشرة من النهاية الطرفية المركزية هى درجات مادة واحدة لكل طالب . وباقى البيانات كانت محولة للنهاية الطرفية المركزية من ملف البيانات أو مولدة بواسطة البرنامج (البيانات المدخلة موضوع تحتها خط) .

```
COURSE:COMP SCI 100          TERM:FALL, 1982

ADAMS B F       SCORE= ?75
AVERAGE= 71.6667

BROWN P         SCORE= ?80
AVERAGE= 65

DAVIS R A       SCORE= ?35
AVERAGE= 40

FISHER E K      SCORE= ?5
AVERAGE= 4.16667

HAMILTON S P    SCORE= ?90
AVERAGE= 91.6667

JONES J J       SCORE= ?80
AVERAGE= 87.5

LUDWIG C W      SCORE= ?70
AVERAGE= 53.3333

OSBORNE T       SCORE= ?80
AVERAGE= 70

PRINCE W F      SCORE= ?100
AVERAGE= 82.5

RICHARDS E N    SCORE= ?70
AVERAGE= 55

SMITH M C       SCORE= ?75
AVERAGE= 67.5

THOMAS B A      SCORE= ?10
AVERAGE= 21.6667

WOLFE H         SCORE= ?95
AVERAGE= 64.1667

ZORBA D R       SCORE= ?95
AVERAGE= 78.3333
```

شكل ٨ — ٣

كتابة ملف بيانات متتابع Writing a Sequential Data File

يمكن كتابة المعلومات على ملف البيانات بواسطة برنامج بيسك ، بطريقة تشبه قراءة البيانات من ملف البيانات . عند الكتابة لملف بيانات متتابع ، توضع المعلومات الجديدة أتوماتيكياً وراء البيانات الموجودة ، وبذلك تحمى البيانات الموجودة فعلا فى الملف . إذا كان من الواجب إلغاء البيانات القديمة قبل كتابة بيانات جديدة ، فيجب أن يلغى الملف بصراحة ويعاد وضعه إلى نقطة بدايته .

سوف نرى فى المثال التالى كيف يمكن أن تقرأ البيانات من ملف بيانات متتابع ومن الكونسول ، ثم تشغل وتكتب النتائج بعد ذلك على الكونسول وعلى ملف بيانات متتابع جديد ، ولإنجاز ذلك سوف نستفيد بجمل التعامل مع الملف وهى FILES و QUOTE و SCRATCH و INPUT و PRINT و IF END . وسوف تتم مناقشة الغرض من كل جملة فى المثال .

مثال ٨ — ٤ تشغيل درجات اختبارات طالب Processing Student Examination Scores

سوف نستكمل فى هذا المثال المسألة الموصوفة فى الأمثلة ٨ — ٢ و ٨ — ٣ ، والتى تتناول تسجيل وتشغيل مجموعة من درجات اختبار فصل من الطلبة . دعنا الآن نطور البرنامج حتى يمكنه بصورة أعم تأدية العمليات التالية لكل طالب :

١ — يقرأ مجموعة من درجات اختبار من ملف بيانات موجود (مثال SCORES) .

٢ — يدخل درجة اختبار جديدة من النهاية الطرفية المركزية .

٣ – يحسب متوسطاً لكل درجات الاختبارات .

٤ – يطبع المتوسط المحسوب على النهاية الطرفية المركزية .

٥ – يكتب مجموعة جديدة من البيانات (أى درجات الاختبارات الأساسية ودرجة الاختبار الجديدة والمتوسط المحسوب) على ملف بيانات جديد .

عند إستكمال الحسابات فإن ملف البيانات الجديد سوف يحتوى على كل المعلومات الموجودة فى ملف البيانات الأصلى ، علاوة على درجة الاختبار الإضافية ومتوسط الدرجات لكل طالب .

طريقة إجراء الحسابات Computational Procedure

سوف نكتب البرنامج بطريقة تسمح لنا « إذا رغبنا فى ذلك » . بإضافة درجة اختبار جديدة لكل طالب بدون حساب المتوسط وذلك يمكننا من استخدام البرنامج أثناء الفصل الدراسى لتسجيل بيانات إضافية (درجات اختبار) وعند نهاية الفصل الدراسى يسمح بتشغيل هذه البيانات (أى حساب متوسط الدرجات التى تستخدم لتقرر درجة الطالب النهائية) .

وحتى يكون البرنامج برنامجاً عاماً بقدر المستطاع ، دعنا ندخل كل درجات الاختبار للطالب فى المجموعة المتراصة C .وبذلك فإن C(1) سوف تشير إلى درجة الاختبار الأولى و C(2) لدرجة الاختبار الثانية و ..إلخ . وسوف نسمح بعدد من الدرجات لكل طالب تصل إلى 15 درجة اختبار .

وسوف يشير المتغير K لاختبار معين يتم إدخاله من الكونسول على سبيل المثال . إذا كانت الدرجات التى ندخلها هى للاختبار السادس ، فإننا نحدد قيمة K بعد ذلك بالرقم 6 وكذلك فإن J$ سوف تمثل إما "YES" أو "NO" ويتوقف ذلك على كون حساب متوسط درجات الطالب قد تم أم لا ، وباقى المتغيرات سوف يكون لها نفس التعاريف كما فى المثال ٨ – ٣ .

وسوف تواصل الحسابات كما يلى :

١ – حدد للملف البيانات SCORES قناة رقم 1 ، وللملف UPDATE قناة رقم 2 (وسوف يكون SCORES ملف إدخال UPDATE ملف إخراج) .

٢ – إلغ أى بيانات قديمة يمكن أن تظهر على الملف UPDATE وأعد وضع الملف إلى نقطة البداية وذلك إستعداداً للكتابة عليه .

٣ – اقرأ رقم السطر (N) وعنوان المادة الدراسية (T$) والفصل الدراسى (Y$) من الملف SCORES .

٤ – اطبع عنوان المادة والفصل الدراسى على الكونسول .

٥ – اطبع رقم السطر وعنوان المادة والفصل الدراسى على UPDATE .

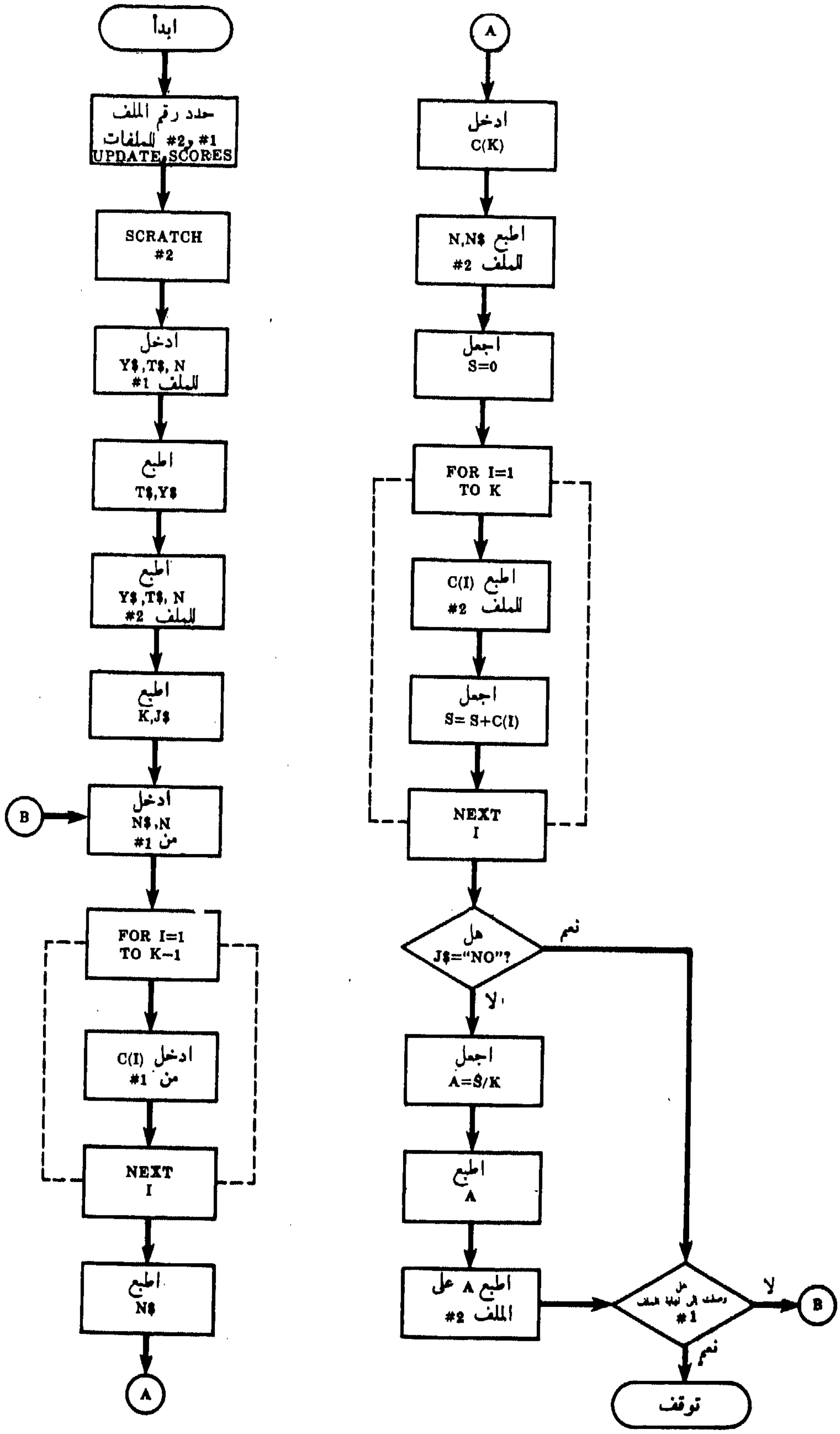
٦ – ادخل قيمة لـ K (رقم الاختبار) من الكونسول .

٧ – أدخل قيمة للمتغير J$ (أما « نعم » "YES" أو « لا » "NO") من الكونسول ، مبيناً إذا كان المطلوب حساب المتوسط لكل طالب أم لا .

٨ – اقرأ رقم سطر (N) واسم طالب (N$) وعدد درجات K — 1 من الملف .

٩ – أدخل درجة الاختبار التى رقمها K من الكونسول .

١٠– اكتب رقم السطر ، واسم الطالب ، ودرجات الطالب كلها وعددها K للملف UPDATE .


ابدأ
حدد رقم الملف
#1 و #2 للملفات
UPDATE, SCORES
SCRATCH
#2
ادخل
Y$,T$, N
للملف #1
اطبع
T$,Y$
اطبع
Y$,T$, N
للملف #2
اطبع
K,J$
B
ادخل
N$,N
من #1
FOR I=1
TO K-1
ادخل C(I)
من #1
NEXT
I
اطبع
N$
A
A
ادخل
C(K)
اطبع N,N$
للملف #2
اجعل
S=0
FOR I=1
TO K
اطبع C(I)
للملف #2
اجعل
S= S+C(I)
NEXT
I
هل
J$="NO"?
نعم
لا
اجعل
A=S/K
اطبع
A
اطبع A على
الملف #2
لا
B
نعم
توقف

شكل ٨ — ٤

١١- إذا كانت "J$="NO تقدم لخطوة ١٣ التالية . وإلا ، احسب متوسط درجات الاختبار (A) باستخدام الصيغة :

$$A = \frac{C(1)+C(2)+\cdots+C(K)}{K}$$

١٢ - اطبع متوسط درجات الاختبار على النهاية الطرفية المركزية واكتب هذه القيمة أيضاً على الملف UPDATE

١٣ - اختبر لترى ما إذا كان هناك بيانات إضافية فى الملف SCORES . إذا كان ذلك صحيحاً ، فارجع مرة أخرى لخطوة ٨ ، وإلا أنه الحسابات .

خريطة سير العمليات المناظرة لذلك مبينة فى شكل ٨ - ٤ .

برنامج بيسك The BASIC Program

يحتوى شكل ٨ - ٥ على برنامج البيسك الحقيقى . لاحظ تضمين عدة جمل للتعامل مع الملفات فى البرنامج .

تحدد جملة FILES الموجودة فى السطر 50 لملفات البيانات SCORES و UPDATE قناتى البيانات 1 و 2 على الترتيب . يجب أن تسبق هذه الجملة أى جملة من جمل الملفات . وبمجرد إعطاء الملفات أرقاماً كقنوات للبيانات المناظرة ، فإن الرجوع لهذه الملفات سوف يكون بواسطة أرقام القنوات وليس بأسماء هذه الملفات .

تحدد الجملة رقم 60 (QUOTE) إن كل السلاسل الحرفية التى تكتب على UPDATE سوف تكون محصورة بين علامتى اقتباس . ويكون ذلك مهماً إذا أردنا قراءة هذه السلاسل الحرفية ببرنامج بيسك فى أى وقت لاحق . يترتب على جملة SCRATCH (سطر رقم 70) مسح أى بيانات سبق تخزينها على الملف UPDATE وإعادته إلى نقطة البداية . ويكون جاهزاً لكتابة أى بيانات .

تقرأ البيانات من الملف SCORES فى أماكن متعددة من البرنامج (السطور 80 و 180 و 200) . وبالمثل فإن البيانات تكتب على الملف UPDATE فى أماكن متعددة (السطور100 و 240 و 270 و 330 و 340) . لاحظ أن أرقام السطور تقرأ وتكتب ، بالرغم من عدم استخدامها فى البرنامج . ونرى أيضاً أن المتغيرات التى تظهر فى جمل PRINT #2 تتبعها الفاصلة المنقوطة (;) . ويترتب على ذلك أن تكون بيانات كل سطر من الملف UPDATE متقاربة بقدر الإمكان .

```
10 REM PROGRAM TO PROCESS STUDENT EXAMINATION SCORES
20 REM       USING SEQUENTIAL DATA FILES
30
40 DIM C(15)
50 FILES SCORES, UPDATE
60 QUOTE #2
70 SCRATCH #2
80 INPUT #1,N,T$,Y$
90 PRINT "COURSE:";T$,"TERM:";Y$
100 PRINT #2,N;T$;Y$
110 PRINT
120 PRINT "EXAM NUMBER";
130 INPUT K
140 PRINT
150 PRINT "CALCULATE AVERAGES (YES OR NO)";
160 INPUT J$
170 PRINT
180 INPUT #1,N,N$
190 FOR I=1 TO K-1
200     INPUT #1,C(I)
210 NEXT I
220 PRINT N$,"SCORE=";
230 INPUT C(K)
240 PRINT #2,N;N$;
250 LET S=0
260 FOR I=1 TO K
270     PRINT #2,C(I);
280     LET S=S+C(I)
290 NEXT I
300 IF J$="NO" THEN 340
310 LET A=S/K
320 PRINT "AVERAGE=";A
330 PRINT #2,A;
340 PRINT #2
350 PRINT
360 IF END #1, THEN 380
370 GOTO 180
380 END
```

شكل ٨ - ٥

وأخيراً نفحص فى السطر 360 هل وصلنا لنهاية الملف SCORES ويتحول التحكم للسطر 380 (END) إذا وجدت نهاية الملف ، وبذلك ننهى الحسابات . وإلا فإن التحكم يتحول إلى الجملة التى تليها فى التنفيذ السطر 370 ، وذلك لتشغيل درجات الطالب التالى .

تطبيق البرنامج Application of the Program

نرى فى شكل ٨ – ٦ كيف يمكن استخدام البرنامج لإضافة درجة اختبار إضافية لكل طالب ، وبدون حساب المتوسط . كل درجة اختبار جديدة تدخل من خلال النهاية الطرفية المركزية (استجابات المستخدم موضوع تحتها خط) .

يحتوى ملف البيانات الجديد (UPDATE) درجات اختبارات . ذلك مبين فى نهاية شكل ٨ – ٦ . وملف البيانات الجديد يولد من ملف البيانات الأصلى SCORES ، وهو مبين فى شكل ٨ – ١ . ويستخدم أمر النظام LIST لإصدار قائمة بملف البيانات UPDATE على الكونسول ، بعد الانتهاء من تنفيذ البرنامج .

```
COURSE:COMP SCI 100          TERM:FALL, 1982
EXAM NUMBER ?6
CALCULATE AVERAGES (YES OR NO) ?NO
ADAMS B F       SCORE= ?75
BROWN P         SCORE= ?80
DAVIS R A       SCORE= ?55
FISHER E K      SCORE= ?5
HAMILTON S P    SCORE= ?90
JONES J J       SCORE= ?80
LUDWIG C W      SCORE= ?70
OSBORNE T       SCORE= ?80
PRINCE W F      SCORE= ?100
RICHARDS E N    SCORE= ?70
SMITH M C       SCORE= ?75
THOMAS B A      SCORE= ?10
WOLFE H         SCORE= ?95
ZORBA D R       SCORE= ?95

TIME:  0.80 SECS.
>OLD
OLD FILE NAME-->UPDATE
>LIST

UPDATE

10  "COMP SCI 100" "FALL, 1982"
20  "ADAMS B F" 45  80  80  95  55  75
30  "BROWN P" 60  50  70  75  55  80
40  "DAVIS R A" 40  30  10  45  60  55
50  "FISHER E K" 0  5  5  0  10  5
60  "HAMILTON S P" 90  85  100  95  90  90
70  "JONES J J" 95  90  80  95  85  80
80  "LUDWIG C W" 35  50  55  65  45  70
90  "OSBORNE T" 75  60  75  60  70  80
100  "PRINCE W F" 85  75  60  85  90  100
110  "RICHARDS E N" 50  60  50  35  65  70
120  "SMITH M C" 70  60  75  70  55  75
130  "THOMAS B A" 10  25  35  20  30  10
140  "WOLFE H" 25  40  65  75  85  95
150  "ZORBA D R" 65  80  70  100  60  95
```

شكل ٨ – ٦

نفرض أننا أردنا أن نستخدم ملف البيانات الجديد فى وقت لاحق كملف إدخال لنفس برامج البيسك (BASIC) ولذلك فن المهم أن نغير اسم ملف البيانات من UPDATE إلى SCORES ويمكن إنجاز ذلك باستخدام أوامر النظام SCRATCH (وذلك لإزالة ملف البيانات الأصلى SCORES) ثم الأمر RENAME (وذلك لتغير اسم UPDATE إلى SCORES) . والإجراء مبين

```
>OLD
OLD FILE NAME-->SCORES
>SCRATCH
>OLD
OLD FILE NAME-->UPDATE
>RENAME SCORES
```

شكل ٨ — ٧

فى شكل ٨ — ٧ . (لا تخلط أمر نظام البيسك SCRATCH مع جملة التعامل مع الملف SCRATCH ، على سبيل المثال SCRATCH#2) .

والآن دعنا نرى كيف نستخدم هذا البرنامج لإدخال درجة اختبار جديدة ثم بعد ذلك حساب متوسط الدرجات لكل طالب . درجات الاختبار الجديدة والمتوسطات المحسوبة مبينة فى شكل ٨ — ٨ . (هذا الخرج مولد باستخدام ملف البيانات الأصلى SCORES المبين فى شكل ٨ — ١) .

لاحظ التماثل بين شكل ٨ — ٨ وشكل ٨ — ٣ (والأخير مولد باستخدام البرنامج فى المثال ٨ — ٣) . ولكننا الآن ، ليس عندنا فقط سجل لمتوسط درجات اختبار لكل طالب مبين على النهاية الطرفية المركزية ، ولكن سجل كامل لدرجات الاختبار والمتوسطات فى ملف البيانات UPDATE . قائمة بالملف UPDATE مبينة فى نهاية الشكل ٨ — ٨ .

ملفات البيانات التسلسلية مفيدة عامة لتخزين مجموعات من البيانات التى سوف تشغل بواسطة برنامج بيسك فى أى وقت لاحق . وهناك سببان لذلك . أولا ، أنه يمكن تنقيح ملف بيانات تسلسل باستخدام أوامر نظام بيسك بدون تغير البرنامج . وعلاوة على ذلك ، فن الممكن تضمين كل من البيانات الحرفية والرقية فى ملف بيانات تسلسل .

ومن الناحية الأخرى ، فبعض التطبيقات تتطلب تحويل المعلومات من وإلى ملف البيانات بدون اعتبار للترتيب الذى تخزن به البيانات . واستخدام ملف بيانات تسلسل غير فعال نسبياً فى مثل هذه المواقف ، حيث يمكن تضييع وقت الحاسب الجدير بالاعتبار بالبحث تكراراً فى ملف البيانات على المعلومات المطلوبة . وفى الجزء التالى سوف ندرس نوعاً مختلفاً من ملفات البيانات أكثر ملاءمة للتطبيقات من النوع التسلسلى .

٨ — ٢ ملفات البيانات العشوائية RANDOM DATA FILES

بينما يحتوى ملف البيانات التسلسلية على مجموعات من البيانات مرتبة على أساس تزايد أرقام السطور ، فإن **ملف البيانات العشوائى** يحتوى على بنود بيانات فردية وغير مرتبة بأى نظام معين . كل بند من بنود البيانات يمكن قراءته أو كتابته مباشرة من أو إلى ملف بيانات عشوائى بدون مواصلة متتالية فى ملف البيانات من البداية . ولذلك فإن تحويل المعلومات ومن وإلى ملف البيانات العشوائى أسرع من ملف البيانات التسلسلية .

يمكن أن تتكون ملفات البيانات العشوائية من بيانات رقية أو حرفية ولكن ليس الاثنان معاً . يوصف نوع الملف (إما رقى أو حرفى) بواسطة وضع علامة النسبة المئوية (%) أو علامة الدولار ($) بعد اسم الملف . تعنى علامة % أن الملف رقى ، وعلامة ($) على أن الملف حرفى . ويجب أن تتبع علامة ($) كية صحيحة موجبة (تتراوح من 1 إلى 132) هذه الكمية تحدد العدد الأقصى من الحروف التى يمكن أن تظهر فى كل سلسلة حرفية .

مثال ٨ — ٥

برنامج بيسك يحتوى على الجملة :

```
10 FILES SALES%,MASTER%,ITEMS$20,CUSTMR$72
```

```
COURSE:COMP SCI 100          TERM:FALL, 1982

EXAM NUMBER ?6

CALCULATE AVERAGES (YES OR NO) ?YES

ADAMS B F       SCORE= ?75
AVERAGE= 71.6667

BROWN P         SCORE= ?80
AVERAGE= 65

DAVIS R A       SCORE= ?55
AVERAGE= 40

FISHER E K      SCORE= ?5
AVERAGE= 4.16667

HAMILTON S P    SCORE= ?90
AVERAGE= 91.6667

JONES J J       SCORE= ?80
AVERAGE= 87.5

LUDWIG C W      SCORE= ?70
AVERAGE= 53.3333

OSBORNE T       SCORE= ?80
AVERAGE= 70

PRINCE W F      SCORE= ?100
AVERAGE= 82.5

RICHARDS E N    SCORE= ?70
AVERAGE= 55

SMITH M C       SCORE= ?75
AVERAGE= 67.5

THOMAS B A      SCORE= ?10
AVERAGE= 21.6667

WOLFE H         SCORE= ?95
AVERAGE= 64.1667

ZORBA D R       SCORE= ?95
AVERAGE= 78.3333

TIME:  0.92 SECS.
>OLD
OLD FILE NAME-->UPDATE
>LIST

UPDATE

10   "COMP SCI 100" "FALL, 1982"
20   "ADAMS B F" 45  80  80  95  55  75  71.6667
30   "BROWN P" 60  50  70  75  55  80  65
40   "DAVIS R A" 40  30  10  45  60  55  40
50   "FISHER E K" 0  5  5  0  10  5  4.16667
60   "HAMILTON S P" 90  85  100  95  90  90  91.6667
70   "JONES J J" 95  90  80  95  85  80  87.5
80   "LUDWIG C W" 35  50  55  65  45  70  53.3333
90   "OSBORNE T" 75  60  75  60  70  80  70
100   "PRINCE W F" 85  75  60  85  90  100  82.5
110   "RICHARDS E N" 50  60  50  35  65  70  55
120   "SMITH M C" 70  60  75  70  55  75  67.5
130   "THOMAS B A" 10  25  35  20  30  10  21.6667
140   "WOLFE H" 25  40  65  75  85  95  64.1667
150   "ZORBA D R" 65  80  70  100  60  95  78.3333
```

شكل ٨ — ٨

تم تحديد قنوات البيانات 1 و 2 للملفات SALES و MASTER وهي ملفات بيانات عشوائية رقية ، كما تشير إلى ذلك (%) علامة النسبة المئوية التي تتبع اسم كل ملف . تم تحديد قنوات البيانات 3 و 4 للملفات ITEM و CUSTMR وهي ملفات بيانات عشوائية حرفية حيث أقصى طول للسلسلة الحرفية هو 20 و 72 حرفاً على الترتيب .

لاحظ أن علامة (%) النسبة المئوية أو علامة ($) الدولار يتبعها رقم صحيح موجب ليست جزءاً من اسم الملف الحقيق . ولكنها ملحق بالاسم (تذكر أن كل اسم من أسماء الملفات بدون الملاحق لا يمكن أن يتعدى 6 حروف) .

وملفات البيانات العشوائية على عكس ملفات البيانات المتتالية ، لا يمكن أن نعمل لها قائمة مباشرة على الكونسول . ولا يمكن أن تنقح باستخدام أوامر نظام البيسك . بينما يمكن بسهولة كتابة برنامج بيسك يقوم بأحد هذه المهام أو كليهما . وسوف ترى كيف يمكن إنجاز ذلك فى فصل لاحق . أولا ، يجب أن ندرس كيف يمكن التوصل لكل بند منفرد من البنود الموجودة فى ملف البيانات العشوائى .

تحكم المؤشر Pointer Control

بالرغم من أن بنود البيانات فى ملف البيانات العشوائى ليست منظمة بأى ترتيب . إلا أن أماكن بنود البيانات مرقمة بترقيم متتالٍ من بداية الملف (مبتدءاً بالمكان رقم I ويزاد بمقدار الوحدة لكل بند بيانات متعاقب) . ويستخدم **مؤشر** للإشارة إلى مكان كل بند منفرد من البيانات . ويجب أن يكون المؤشر دائماً موضوعاً فى الوضع الصحيح قبل نقل أى بند من البيانات من أو إلى ملف البيانات .

مثال ٨ — ٦

يحتوى ملف بيانات عشوائى على ثمانى قيم رقية منظمة بالترتيب المبين فيما يلى :

الموضع	القيمة
1	433
2	256
3	307
4	180
5	75
6	224
7	609
8	52

نفرض أننا نريد أن نقرأ بنود البيانات الخامس والثانى والسابع بهذا الترتيب . أولا نضع المؤشر فى الموضع رقم 5 ونقرأ القيمة 75 . ثم نعيد وضع المؤشر فى الموضع رقم 2 ثم نقرأ القيمة 256 . وأخيراً ، نحرك المؤشر إلى الموضع رقم 7 ونقرأ القيمة 609

وسوف يتقدم المؤشر أوتوماتيكياً خطوة واحدة فى كل مرة يقرأ فيها بنداً أو نكتب بنداً من الملف . وبذلك فن الممكن أن نقرأ أو نكتب بنوداً متتالية من ملف بيانات عشوائى . ويمكننا إعادة وضع المؤشر وقتما نريد ، وذلك بواسطة جملة SET (وفى بعض نسخ بيسك ، جملة RESET) ، وتسمح لنا هذه الجملة بالتوصل لبنود البيانات بأى ترتيب نريده .

ومقترناً بجملة SET الدالتين المكتبيتين LOC و LOF . تسمح لنا دالة LOC بتحديد موضع المؤشر ، وتشير دالة LOF لآخر مكان تخزين فى الملف . والمثال التالى يوضح استخدام جملة SET ودالتى LOC و LOF .

مثال ٨ — ٧

فيما يلى بناء هيكل لبرنامج بيسك يتعامل مع بيانات من ملف بيانات عشوائى رقمى .

```
10 FILES VALUES%
   ...
40 SET :1,K
   ...
80 IF LOC(1)>LOF(1) THEN 200
   ...
110 SET :1,LOC(1)+3
   ...
150 SET :1,(1+LOF(1))/2
```

تحدد جملة 15 قناة البيانات 1 لملف البيانات العشوائى الرقمى VALUES . وفى الجملة رقم 40 يحدد موضع المؤشر لقناة البيانات 1 بالمكان المشار إليه بواسطة المتغير K (نفرض أن K لها قيمة صحيحة موجبة) . وتسبب الجملة رقم 80 تحويل التحكم لجملة رقم 200 إذا كان المؤشر الخاص بقناة البيانات 1 موضوعاً فى أى مكان وراء نطاق نهاية الملف .

وفى الجملة رقم 110 نعيد وضع المؤشر لقناة البيانات 1 ثلاثة أماكن وراء نطاق الموضع الحالى له . وأخيراً ، فى الجملة 150 نعيد وضع المؤشر إلى نقطة المنتصف فى الملف ، محدداً كمتوسط لأول وآخر موقع فى الملف .

لاحظ أن مكان المؤشر فى الجمل 110 و 150 تم تحديدها بواسطة الصيغة 3 + (1) LOC و 2/ ((1) LOF + 1) بالترتيب . مسموح بالصيغة الرياضية فى جملة SET على أن تكون قيمتها موجبة ولكن لا تتعدى آخر موضع فى الملف . وسوف تبتر القيم غير الصحيحة أتوماتيكياً .

ولاحظ أيضاً ، أن رقم القناة فى جملة SET تسبقها علامة الوقف الاستدراكى (:) بدلا من علامة الجنيه #. وبهذه الطريقة نميز ملف البيانات العشوائى عن ملف البيانات التسلسلى . كل جمل التعامل مع الملفات والتى تشير إلى رقم القناة تستعمل هذه العلامة .

وسوف نرى برنامج بيسك كاملا يستخدم جملة SET والدالتين LOC و LOF فى المثال ٨ – ١٣ .

قراءة ملف بيانات عشوائى Reading a Random Data File

لقد رأينا أن ملف البيانات العشوائى يمكن قراءته على التوالى أو عشوائياً . وإذا أردنا أن نقرأ البيانات على التوالى فيمكن إغفال موضع المؤشر ، حيث أن جملة FILES تضع المؤشر عند أول موضع فى الملف وفى كل مرة يقرأ بند من البيانات يتقدم المؤشر أتوماتيكياً إلى الموضع التالى مباشرة .

مثال ٨ – ٨

يحتوى ملف البيانات العشوائى STATES على أسماء 50 ولاية فى الولايات المتحدة الأمريكية (.U.S.A) ومرتبة هجائياً . يمثل شكل ٨ – ٩ برنامج بيسك يقرأ اسم كل ولاية من ملف البيانات ثم يطبع الاسم على الكونسول . وسوف تقرأ البيانات على التوالى وبنفس الترتيب التى تم تخزينها به فى الملف .

تنجز جملة FILES عدة أشياء فى هذا البرنامج . أولا ، تحدد لملف البيانات العشوائى للسلاسل الحرفية STATES قناة البيانات رقم 1 ، وتصف كل سلسلة حرفية بعدد من الحروف لا يتجاوز 15 حرفاً .

لاحظ أننا نستعمل جملة التعامل مع الملف READ وليس INPUT عند القراءة من ملف البيانات العشوائى . (فى عديد من جمل البيسك تستخدم جمل READ و WRITE لملفات البيانات العشوائية وجمل INPUT و PRINT لملفات البيانات التسلسلية) .

```
10 FILES STATES$15
20 FOR I=1 TO 50
30     READ :1,N$
40     PRINT N$
50 NEXT I
60 END

>RUN

FILGEN          18:52          23-MAR

ALABAMA
ALASKA
ARIZONA
ARKANSAS
CALIFORNIA
COLORADO
CONNECTICUT
DELAWARE
FLORIDA
GEORGIA
HAWAII
IDAHO
ILLINOIS
INDIANA
IOWA
KANSAS
KENTUCKY
LOUISIANA
MAINE
MARYLAND
MASSACHUSETTS
MICHIGAN
MINNESOTA
MISSISSIPPI
MISSOURI
MONTANA
NEBRASKA
NEVADA
NEW HAMPSHIRE
NEW JERSEY
NEW MEXICO
NEW YORK
NORTH CAROLINA
NORTH DAKOTA
OHIO
OKLAHOMA
OREGON
PENNSYLVANIA
RHODE ISLAND
SOUTH CAROLINA
SOUTH DAKOTA
TENNESSEE
TEXAS
UTAH
VERMONT
VIRGINIA
WASHINGTON
WEST VIRGINIA
WISCONSIN
WYOMING

TIME:  0.08 SECS.
```

شكل ٨ — ٩

الخرج الناتج من هذا البرنامج (أى قائمة بملف البيانات) مبين عند نهاية شكل ٨ — ٩ .

عند قراءة البيانات الموجودة فى الملف عشوائياً . يجب أن نحرك المؤشر إلى المكان الصحيح قبل قراءة أى بند . ونستخدم جملة SET لهذا الغرض ، كما هو مبين فى المثال التالى — مثال ٨ — ٩ .

مثال ٨ — ٩

دعنا ندرس مرة أخرى ملف البيانات العشوائى STATES ، كما وصفناه فى المثال ٨ — ٨ . ونرغب الآن فى كتابة برنامج بسيط يؤدى الخطوات التالية :

١ – يدخل من الكونسول . موضع السلسلة الحرفية فى الملف STATES (كل موضع يمكن توصيفه بواسطة رقم صحيح ثابت له قيمة تتراوح ما بين 1 إلى 50) .

٢ – تقرأ السلسلة الحرفية (أى اسم الولاية) الموجودة فى هذا الموضع .

٣ – تطبع السلسلة الحرفية على الكونسول .

يتكرر هذا الإجراء حتى يتحدد موضع له قيمة أقل من 1 أو أكبر من 50 .

يمثل الشكل ٨ – ١٠ برنامج البيسك المطلوب . وتمثل المتغيرات L و N$ الموضع المطلوب والسلسلة الحرفية الموجودة فى هذا الموضع ، على الترتيب . لاحظ أن جملة SET تسبق جملة READ ، وبذلك يوضع المؤشر عند الموضع الصحيح قبل قراءة كل بند من البيانات .

```
10  FILES STATES#15
20  PRINT "LOCATION";
30  INPUT L
40  IF L<1 THEN 110
50  IF L>50 THEN 110
60  SET :1,L
70  READ :1,N$
80  PRINT N$
90  PRINT
100 GO TO 20
110 END

>RUN

EX8.9         19:02          23-MAR

LOCATION ?38
PENNSYLVANIA

LOCATION ?23
MINNESOTA

LOCATION ?50
WYOMING

LOCATION ?11
HAWAII

LOCATION ?8
DELAWARE

LOCATION ?45
VERMONT

LOCATION ?0

TIME:  0.31 SECS.
```

شكل ٨ – ١٠

يبين الجزء الأسفل من شكل ٨ – ١٠ الخرج الذى يولد كإجابة لإدخال عدة كيات . لاحظ أن أسماء الولايات تطبع بالترتيب الموصوف بواسطة جملة INPUT وليس بالترتيب التى تخزن به الأسماء فى ملف البيانات .

كتابة ملف بيانات عشوائى Writing a Random Data File

يمكن كتابة بند بيانات على ملف عشوائى بطريقة تشبه طريقة القراءة ، غير أننا نستخدم الآن جملة التعامل مع الملف WRITE بدلاً من READ . وكما ذكرنا سابقاً ، فإن المؤشر يجب وضعه فى المكان الصحيح قبل كتابة بند البيانات . وسوف يحل بند البيانات الجديد محل ما كان مخزناً سابقاً فى هذا الموضع .

مثال ٨ – ١٠ مراقبة المخزون Inventory Control

يحتفظ مستودع بقائمة جرد لعدد معين من السلع . وسوف تتذبذب كية كل سلعة من يوم إلى آخر . وذلك تبعاً للأوامر التى تطلب بواسطة العملاء (وبذلك تقل كية المخزون) وتبعاً لوصول الشحنات من الموردين (وبذلك تزيد كية المخزون) . ونريد الاحتفاظ بسجل بالمخزون المضبوط لكل بند . ولعمل ذلك يجب أن نعدل مستوى المخزون بعد كل معاملة على حدة .

طريقة إجراء الحسابات Computational Procedure

يمكن تسجيل مستوى المخزون لكل سلعة بطريقة ملائمة فى ملف بيانات عشوائى رقمى . وسوف نستعمل ملف البيانات INVTRY لهذا الغرض .

سوف تحدد كل سلعة برقم مخزون فريد (رقم صحيح موجب) يعرف بالسلعة . وسوف يستخدم رقم المخزون أيضاً ليشير إلى موضع سجل المخزون فى ملف البيانات . وبذلك إذا كان المطلوب معرفة كم عدد الوحدات للسلعة رقم 86 الموجودة حالياً فى المستودع فيجب أن نفحص محتويات مكان التخزين رقم 86 فى ملف البيانات .

١ – حدد ملف البيانات INVTRY بقناة البيانات 1 .

٢ – اطبع حجم ملف البيانات (أى قيمة آخر موضع تخزين) على الكونسول .

٣ – ادخل قيمة لرقم المخزون (P) من على الكونسول .

٤ – إذا كانت قيمة P أقل من 1 أو أكبر من آخر موضع فى ملف البيانات ، أنه البرنامج والحسابات ، وألا تقدم للخطوة ٥ التالية .

٥ – ضع مؤشر ملف البيانات فى الموضع P .

٦ – اقرأ مستوى المخزون (N) من ملف البيانات .

٧ – اطبع القيمة الحالية N على الكونسول .

٨ – أدخل التغير فى مستوى المخزون (N1) ، بدلالة عدد الوحدات من الكونسول . (لاحظ نقص مستوى المخزون يجب أن يشار إليه بإدخال قيمة سالبة للمتغير N1)

٩ – احسب مستوى مخزون جديد ، أى ، اجعل :

$$N = N + N1$$

١٠ – إذا كانت قيمة N الجديدة سالبة (وهذا لا يتحقق فى الواقع) فاجعل قيمة N مساوية للصفر .

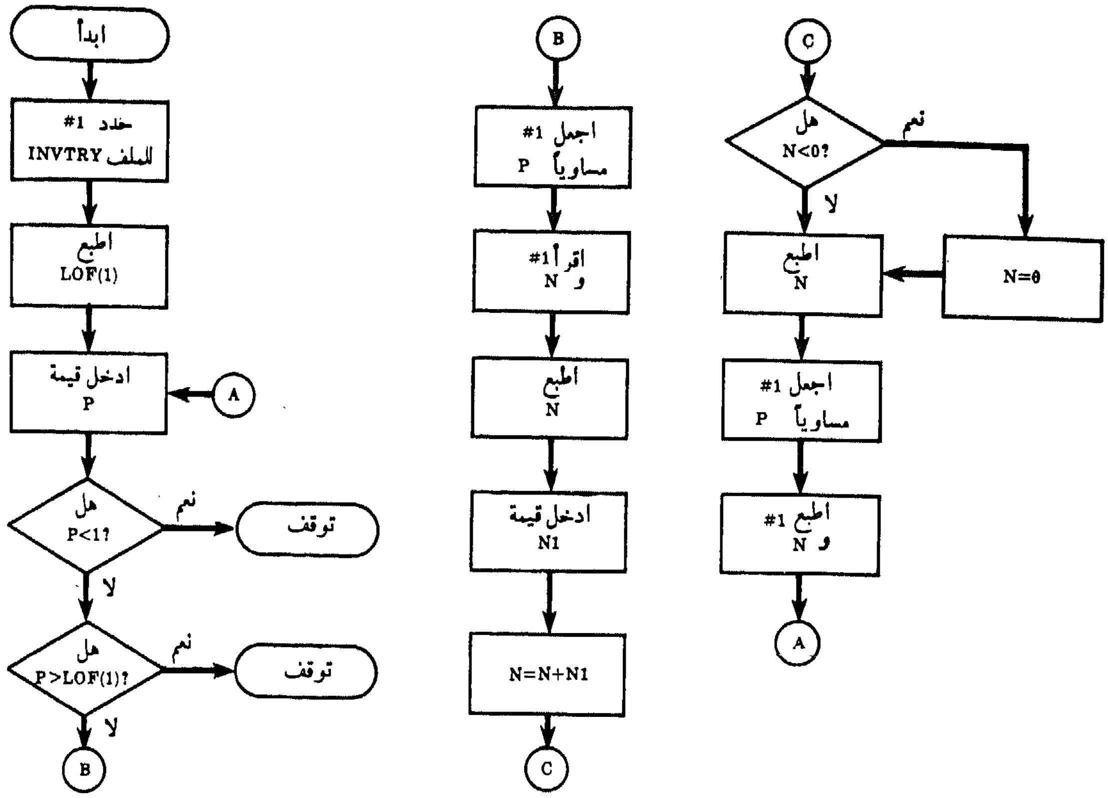
١١ – اطبع القيمة الجديدة N على الكونسول .

١٢ – أعد المؤشر للموقع P (لاحظ أن المؤشر سوف يتقدم للموقع P + 1 بعد الخطوة رقم ٦) .

١٣ – اكتب قيمة N الجديدة على ملف البيانات .

١٤ – ارجع مرة ثانية للخطوة ٣

يبين شكل ٨ – ١١ خريطة سير العمليات المناظرة لذلك .



شكل ٨ – ١١

```
10 REM INVENTORY CONTROL PROGRAM
20 FILES INVTRYX
30 PRINT "STOCK NUMBERS RUN FROM 1 TO ";LOF(1)
40 PRINT
50 PRINT "STOCK NUMBER";
60 INPUT P
70 IF P<1 THEN 220
80 PRINT
90 SET :1,P
100 READ :1,N
110 PRINT "ORIGINAL INVENTORY=";N;" ITEMS"
120 PRINT "CHANGE IN INVENTORY LEVEL";
130 INPUT N1
140 LET N=N+N1
150 IF N>=0 THEN 170
160 LET N=0
170 PRINT "NEW INVENTORY=";N;" ITEMS"
180 PRINT
190 SET :1,P
200 WRITE :1,N
210 GOTO 50
220 END

>RUN

EX8.10          19:04          23-MAR

STOCK NUMBERS RUN FROM 1 TO  2000

STOCK NUMBER ?1186

ORIGINAL INVENTORY= 346  ITEMS
CHANGE IN INVENTORY LEVEL ?-45
NEW INVENTORY= 301  ITEMS

STOCK NUMBER ?708

ORIGINAL INVENTORY= 368  ITEMS
CHANGE IN INVENTORY LEVEL ?200
NEW INVENTORY= 568  ITEMS

STOCK NUMBER ?84

ORIGINAL INVENTORY= 147  ITEMS
CHANGE IN INVENTORY LEVEL ?16
NEW INVENTORY= 163  ITEMS

STOCK NUMBER ?1400

ORIGINAL INVENTORY= 78  ITEMS
CHANGE IN INVENTORY LEVEL ?-50
NEW INVENTORY= 28  ITEMS

STOCK NUMBER ?0

TIME:  0.29 SECS.
```

شكل ٨ – ١٢

برنامج البيسك The BASIC Program

يحتوى شكل ٨ – ١٢ على برنامج بيسك حقيقى . نرى أن البرنامج يحتوى على جمل التعامل مع الملفات FILES و SET و READ و WRITE كذلك الدالة المكتبية LOF . لاحظ أن أى جملة من جمل READ و WRITE دائماً مسبوقة بجملة SET ، وبذلك يتم وضع المؤشر فى المكان الصحيح قبل نقل أى معلومات من أو إلى ملف البيانات .

الجزء الأسفل من شكل ٨ – ١٢ يبين مجموعة بيانات نموذجية ناتجة من تنفيذ البرنامج كذلك يبين كلا من بيانات الإدخال والمخرجات . لاحظ أن تنفيذ البرنامج يتوقف بإدخال قيمة صفر لأحد أرقام المخزون .

يمكننا تضمين عدة بنود للإخراج فى جملة WRITE واحدة إذا أردنا ذلك . فى مثل هذه الحالات فإن أول بند سوف يوضع فى المكان الذى يحدده المؤشر ، وبنود البيانات التالية سوف تخزن فى المواضع المتتالية . كما فى جملة PRINT ، يمكن أن يتمثل كل بند من البيانات بواسطة رقم ثابت أو متغير أو صيغة رياضية أو مرجع دالة .

مثال ٨ – ١١

برنامج بيسك يحتوى على الجمل :

```
10 FILES DATA%
   ...
100 SET :1,P
110 WRITE :1,C1,(A+B)/2,SQR(X)
```

نفرض أن P قيمتها 39 فإن القيمة التى تعطى للمتغير C1 سوف توضع فى المكان 39 فى ملف البيانات الرقية DATA وسوف يحتوى الموضع رقم 40 على القيمة الممثلة بالصيغة الرياضية (A+B)/2 والموضع رقم 41 على القيمة SQR (X)

انشاء ملف بيانات عشوائى Creating a Random Data File

لا يمكن أن ينشأ ملف بيانات عشوائى باستخدام أوامر نظام بيسك ويجب علينا إذن أن نكتب برنامج بيسك لإنشاء ملف بيانات عشوائى . والإجراء لتنفيذ ذلك ممثل فى البرنامج التالى .

مثال ٨ – ١٢

نفرض أننا نرغب فى إنشاء ملف البيانات العشوائى STATES بواسطة برنامج بيسك يسمى FILGEN . برنامج البيسك وأوامر النظام المتزاملة معه مبينة فى شكل ٨ – ١٣ . ومرة أخرى فإن أوامر النظام المضافة بواسطة المستخدم كلها موضوع تحتها خط .

لاحظ أن السطور الثلاثة الأولى تعرف الملف الذى تم الاحتفاظ به STATES ، بالرغم من أن البيانات مازالت لم تدخل إلى STATES وهذا مطلوب حتى يتعرف النظام على جملة FILES التى تعرف اسم الملف (السطر 10)

```
>NEW
NEW FILE NAME-->STATES
>SAVE
>NEW
NEW FILE NAME-->FILGEN
>10 FILES STATES#15
>20 INPUT N$
>30 IF N$="END" THEN 60
>40 WRITE :1,N$
>50 GO TO 20
>60 END
>SAVE
```

شكل ٨ – ١٣

وعند تنفيذ البرنامج FILGEN سوف تدخل مجموعة من السلاسل الحرفية من على الكونسول ونكتبها على STATES ، مبتدءاً بالمكان 1 ويكمل فى مواضع التخزين المتعاقبة وسوف ننهى الحسابات عند طباعة كلمة END بعد دخول كل البيانات .

٨ ـــ ٣ مواصفات ملف اثناء وقت التشغيل RUN TIME FILE SPECIFICATIONS

فى عدة تطبيقات نرغب فى كتابة برنامج بيسك يستخدم ملفات بيانات ولكنه لا يحدد أى أسماء ملفات معينة . ولكن يمكن أن ندخل أسماء الملفات كبيانات إدخال أثناء تنفيذ البرنامج والبرنامج الذى يكتب بهذه الطريقة سوف يكون برنامجاً عاماً أكثر من برنامج يتطلب ملفات بيانات محددة .

من السهل جداً أن ندخل أسماء الملفات المطلوبة أثناء وقت التشغيل (أى أثناء تنفيذ البرنامج) إذا أردنا ذلك . ويجب فى هذه الحالة استخدام جملة FILE وليس جملة FILES (لاحظ أن جملتى FILE و FILES مختلفتان تماماً ، سوف ترى ذلك فى المثال التالى) .

مثال ٨ – ١٣ البحث فى ملف بيانات Searching a Data File

هذا المثال يقدم طريقة فنية فعالة للتوصل لموضع بند بيانات معين فى ملف بيانات عشوائى يحتوى على سلاسل الحروف .وسوف نفترض أن سلاسل الحروف مخزنة فى الملف بترتيب هجائى . وتعرف الطريقة باسم **البحث الثنائى** ، وهى تشبه الشكل الممثل فى المثال ٦ – ٦ لإيجاد أقصى قيمة للدالة .

طريقة إجراء الحسابات Computational Procedure

دعنا ندرس نطاقاً للبحث يتكون من عدة مواضع تخزين متعاقبة فى الملف . مبدئياً سوف يكون نطاق البحث عبارة عن الملف بأكمله . وسوف تكون خطتنا هى مقارنة سلسلة الحروف عند منتصف نطاق البحث مع سلسلة الحروف المطلوبة . سوف نحصل على إحدى ثلاث نتائج .

١ – أن تكون سلسلة الحروف الموجودة عند نقطة المنتصف هى سلسلة الحروف المطلوبة . وفى هذه الحالة سوف يتوقف إجراء الحساب .

٢ – أن تكون سلسلة الحروف المطلوبة فى النصف الأول من نطاق البحث ومن ثم يهمل النصف الثانى من نطاق البحث ، وتقارن سلسلة الحروف المطلوبة سلسلة الحروف عند منتصف النصف الأول من نطاق البحث .

٣ – أن تكون سلسلة الحروف فى النصف الثانى من نطاق البحث . وفى هذه الحالة سوف نلغى النصف الأول من نطاق البحث ، ثم تقارن سلسلة الحروف المطلوبة بسلسلة الحروف الموضوعة فى منتصف الجزء الفرعى المتبقى من النطاق .

يكرر هذا الإجراء حتى يتم العثور على سلسلة الحروف المطلوبة أو نقرر أن السلسلة غير موجودة فى ملف البيانات .

المخطط التمهيدى للبرنامج The Program Outline

من أجل تخطيط الإجراء دعنا نعرف المتغيرات التالية :

F$ = اسم ملف البيانات العشوائى الحرفى ، متضمنا الملحق .

N$ = سلسلة الحروف التى يجب وضعها فى ملف البيانات .

M$ = سلسلة الحروف تقرأ من ملف البيانات وتقارن بـ N$

P1 = مؤشر يشير إلى بداية نطاق البحث .

P2 = مؤشر يشير إلى نقطة منتصف فى نطاق البحث .

P3 = مؤشر يشير إلى نهاية نطاق البحث .

وسوف نواصل الحسابات كما يلي :

١ – يدخل اسم الملف من الكونسول وتعطى للمتغير F$

٢ – يلازم الملف الممثل بواسطة F$ قناة البيانات رقم 1 .

٣ – تدخل سلسلة الحروف من الكونسول وتعطى للمتغير N$. إذا كانت N$=END ، فإن تنفيذ البرنامج سوف يتوقف ، وإلا فسوف نستمر فى الحسابات من الخطوة ٤ التالية .

٤ – تحدد قيم مبدئية مقدارها 1 للمؤشرات P1 و P2 و LOF(1) على الترتيب وذلك يعرف نطاق البحث المبدئى .

٥ – إذا ضيقنا نطاق البحث إلى حد أن P1 و P3 تشير إلى مواضع متجاورة ، فإن قيمة M$ تقرأ من الموضع P1 ثم تقارن بـ N$

(ا) إذا كانت N$=M$ فإن التحكم سوف ينتقل إلى الخطوة ٨ أدناه .

(ب) إذا كانت M$ تختلف عن N$ فتقرأ قيمة أخرى M$ من الموضع P3 ثم تقارن بـ N$. .

(جـ) إذا كانت N$=M$ فإن التحكم سوف ينتقل إلى الخطوة ٨ أدناه .

(د) إذا كانت قيمة M$ لا تساوى N$ ، فتطبع رسالة على الكونسول تشير إلى أن السلسلة الحرفية المطلوبة غير موجودة . ثم نرجع مرة ثانية للخطوة ٣ عاليه .

٦ – إذا لم تشر P1 و P3 إلى مواضع متجاورة ، بذلك يمكن تحديد قيمة للمتغير P2 باستخدام الصيغة الرياضية .

P2=INT((P1+P3)/2)

٧ – تقرأ قيمة للمتغير M$ من الموضع P2 ثم يقارن بـ N$

(ا) إذا كانت N$=M$, فالتحكم سوف ينتقل للخطوة ٨ أدناه .

(ب) إذا كانت N$<M$ ، نحتفظ بنصف نطاق البحث الأول . من ثم تعطى P3 قيمة جديدة تحسب كالتالى

P3=P2−1

ويتحول التحكم مرة ثانية إلى الخطوة رقم ٥ عاليه .

(جـ) إذا كانت N$>M$ ، نحتفظ بنصف نطاق البحث الثانى. من ثم نحسب قيمة جديدة للمتغير P1 كالتالى :

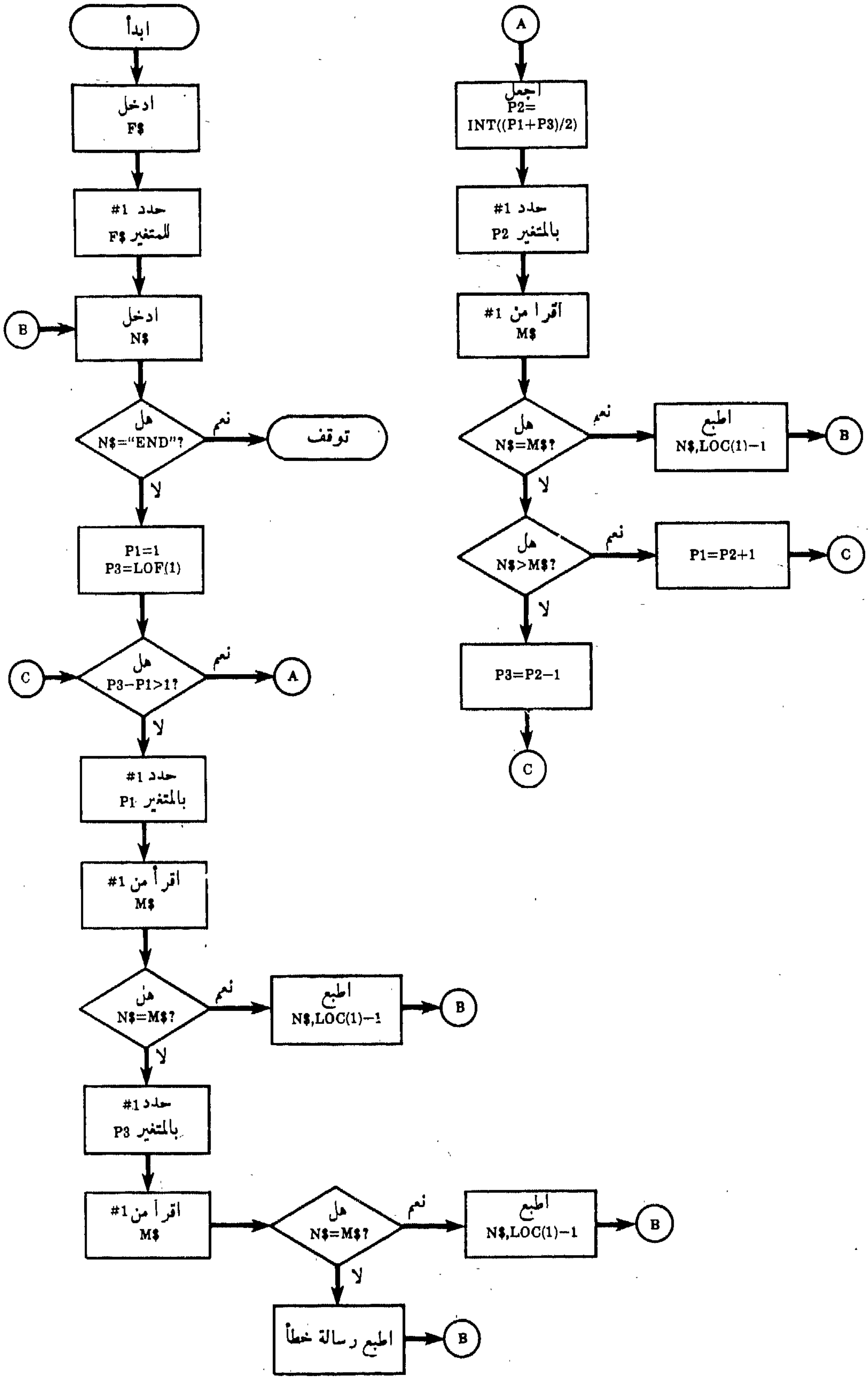
P1=P2+1

ويتحول التحكم مرة ثانية إلى الخطوة رقم ٥ عاليه .

٨ – تطبع رسالة على النهاية الطرفية المركزية تشير إلى أن سلسلة الحروف المطلوبة قد تم تخزينها فى الموضع رقم LOC (1) — 1 (لاحظ أننا استخدمنا الموضع رقم LOC (1) — 1 وليس LOC (1) لأن المؤشر سوف يكون قد تقدم خطوة عند قراءة أحدث قيمة للمتغير M$) .

٩ – بعد ذلك نرجع للخطوة رقم (٣) . وبذلك نعيد البحث عن سلسلة الحروف الجديدة .

خريطة سير العمليات المناظرة لذلك مبينة فى شكل ٨ – ١٤ .


ابدأ
ادخل
F$
حدد #1
للمتغير F$
ادخل
N$
B
هل
N$="END"?
نعم
توقف
لا
P1=1
P3=LOF(1)
C
هل
P3−P1>1?
نعم
A
لا
حدد #1
بالمتغير P1
اقرأ من #1
M$
هل
N$=M$?
نعم
اطبع
N$,LOC(1)−1
B
لا
حدد #1
بالمتغير P3
اقرأ من #1
M$
هل
N$=M$?
نعم
اطبع
N$,LOC(1)−1
B
لا
اطبع رسالة خطأ
B
A
اجعل
P2=
INT((P1+P3)/2)
حدد #1
بالمتغير P2
اقرأ من #1
M$
هل
N$=M$?
نعم
اطبع
N$,LOC(1)−1
B
لا
هل
N$>M$?
نعم
P1=P2+1
C
لا
P3=P2−1
C

شكل ٨ – ١٤

برنامج البيسك The BASIC Program

يحتوى شكل ٨ – ١٥ على برنامج بيسك كامل لهذه المسألة . ولقد تضمن البرنامج استخدام ثلاث جمل للتعامل مع الملف – وهى FILE (فى السطر 40) و SET (فى السطرين 180 و 290) و READ (السطور 190 و 210 و 300) . وتوجد فى البرنامج أيضاً جملتا LOC و LOF فى السطور 460 و 130 على الترتيب . لاحظ أن جملة FILES المعتادة والتى تحدد قنوات بيانات للملفات البيانات ، غير موجودة . فى الواقع ، فالبرنامج يقرأ اسم ملف فى النهاية الطرفية المركزية (سطر 30) ، ثم بعد ذلك يحدد لاسم الملف هذا قناة البيانات 1 بواسطة جملة FILE فى السطر 40 .

```
10 REM          BINARY SEARCH PROCEDURE
20 PRINT "FILE NAME";
30 INPUT F$
40 FILE :1,F$
50
60 REM          ENTER STRING AND ESTABLISH INITIAL SEARCH INTERVAL
70
80 PRINT
90 PRINT "DESIRED STRING";
100 INPUT N$
110 IF N$="END" THEN 480
120 LET P1=1
130 LET P3=LOF(1)
140
150 REM         TEST FOR SMALL INTERVAL
160
170 IF P3-P1>1 THEN 260
180 SET :1,P1
190 READ :1,M$
200 IF N$=M$ THEN 440
210 READ :1,M$
220 IF N$=M$ THEN 440
230 PRINT N$;" IS NOT IN THE DATA FILE"
240 GOTO 60
250
260 REM         LOCATE MIDPOINT AND COMPARE
270
280 LET P2=INT((P1+P3)/2)
290 SET :1,P2
300 READ :1,M$
310 IF N$=M$ THEN 440
320 IF N$>M$ THEN 390
330
340 REM         RETAIN FIRST HALF OF SEARCH INTERVAL
350
360 LET P3=P2-1
370 GOTO 150
380
390 REM         RETAIN LAST HALF OF SEARCH INTERVAL
400
410 LET P1=P2+1
420 GOTO 150
430
440 REM         DESIRED STRING HAS BEEN LOCATED - PRINT OUTPUT
450
460 PRINT N$;" IS STORED IN LOCATION";LOC(1)-1
470 GOTO 60
480 END
```

شكل ٨ – ١٥

الحوار الناتج من تشغيل هذا البرنامج مبين فى شكل ٨ – ١٦ . ونرى أن ملف البيانات STATES والذى تمت مناقشته فى الأمثلة ٨ – ٨ و ٨ – ٩ و ٨ – ١٢ ، هو الذى نرغب فى فحصه فى هذا المثال . لاحظ أن الملحق (أى $15) يتم إدخاله مع اسم الملف (البيانات المدخلة موضوع تحتها خط) .

```
FILE NAME ?STATES#15

DESIRED STRING ?PENNSYLVANIA
PENNSYLVANIA IS STORED IN LOCATION 38

DESIRED STRING ?FLORIDA
FLORIDA IS STORED IN LOCATION 9

DESIRED STRING ?OHIO
OHIO IS STORED IN LOCATION 35

DESIRED STRING ?ALASKA
ALASKA IS STORED IN LOCATION 2

DESIRED STRING ?PUERTO RICO
PUERTO RICO IS NOT IN THE DATA FILE

DESIRED STRING ?CALIFORNIA
CALIFORNIA IS STORED IN LOCATION 5

DESIRED STRING ?MASSACHUSETTS
MASSACHUSETTS IS STORED IN LOCATION 21

DESIRED STRING ?END
TIME:  0.45 SECS.
```

شكل ٨ – ١٦

وأخيراً ، يجب أن نؤكد أن هذا البرنامج ، وعلى عكس البرامج التى تمت مناقشتها فى الأمثلة السابقة ، يمكن تشغيله مع أى ملف بيانات حرفى – لأن ملف البيانات المقصود فحصه يوصف ككمية مدخلة وليس كجزء من البرنامج . وهذه الطريقة فى توصيف الملف تزيد من تعميم البرنامج بدرجة كبيرة .

لم نناقش فى هذا الفصل بعض جمل التعامل مع الملف ، مثل RESTORE و MARGIN و PAGE . (بعض نسخ بيسك تتضمن أيضاً مجموعة جمل التعامل مع ملف المصفوفة). والسبب فى ذلك ، كما ذكرنا من قبل ، هو التغيرات الموجودة فى جمل التعامل مع الملف بين نسخة بيسك وأخرى . والقارئ الذى يرغب فى استخدام ملفات بيانات يجب أن يقرر تماماً طبيعة جمل التعامل مع الملفات المتاحة على الجهاز الخاص به .

اسئلة للمراجعة

Review Questions

٨ – ١ ما هو الملف ؟ ما هى أنواع المعلومات التى يمكن أن يحتويها ملف ؟

٨ – ٢ ما هو الفرق بين ملف بيانات متسلسل وملف بيانات عشوائى ؟

٨ – ٣ ما هى مزايا ملف بيانات متسلسل عند مقارنته بملف بيانات عشوائى ؟

٨ – ٤ ما هى مزايا ملف بيانات عشوائى عند مقارنته بملف بيانات متسلسل ؟

٨ – ٥ كيف ترتب محاميع البيانات فى ملف بيانات عشوائى ؟

٨ – ٦ هل يمكن دمج الأرقام وسلاسل الحروف فى ملف بيانات متسلسل ؟ ملف بيانات عشوائى ؟

٨ – ٧ كيف يمكن فصل بنود البيانات الموجودة فى ملف بيانات متسلسل عن بعضها ؟ وما هى القواعد الخاصة التى يمكن تطبيقها على سلاسل الحروف فاصلة (,) أو مكان خال ؟

٨ – ٨ هل يمكن لأوامر نظام بيسك أن تستخدم فى إنشاء وتنقيح ملف بيانات متسلسل ؟ ملف بيانات عشوائى ؟

٨ – ٩ ما هى القواعد التى تطبق عند تسمية ملف بيانات ؟

٨ – ١٠ لأى نوع من الملفات يجب أن يتبع اسم الملف ملحق ؟ وما هى المعلومات التى يمدنا بها الملحق ؟

٨ – ١١ هل يجب أن تقرأ بنود البيانات الموجودة فى ملف بيانات متسلسل بترتيب معين ؟

٨ – ١٢ كيف يمكن اختيار نهاية ملف بيانات متسلسل ؟

٨ – ١٣ ما هى قناة البيانات ؟ وكيف يمكن تحديد قناة البيانات لملف البيانات ؟

٨ – ١٤ أين يجب كتابة معلومة جديدة على ملف بيانات عشوائى ؟

٨ – ١٥ كيف يمكن مسح البيانات الموجودة على ملف بيانات متسلسل وإعادته إلى نقطة بدايته ؟

٨ – ١٦ كيف يمكن التحكم فى التباعد بين بنود البيانات الفردية ، عند كتابة مجموعة من البيانات على ملف بيانات متسلسل ؟

٨ – ١٧ كيف يمكن إعادة تسمية ملف بيانات متسلسل ؟

٨ – ١٨ كيف يمكن تنظيم بنود البيانات فى ملف بيانات عشوائى ؟ وكيف يمكن التوصل إلى بند معين من بنود البيانات ؟

٨ – ١٩ ما هو المؤشر ؟ وكيف يمكن إنشاء مكان المؤشر ؟ وكيف يمكن إعادة تحديد وضعه ؟

٨ – ٢٠ كيف تقرأ بنود البيانات من ملف بيانات عشوائى ؟ وهل يمكن قراءة ملف بيانات عشوائى بطريقة متتالية ؟ اشرح.

٨ – ٢١ كيف تكتب بنود البيانات على ملف بيانات عشوائى ؟ هل يمكن كتابة بيانات على ملف بيانات عشوائى بطريقة متتالية ؟ اشرح .

٨ – ٢٢ عند كتابة بند من بنود البيانات على ملف بيانات عشوائى ، ماذا يحدث للمعلومات التى سبق تخزينها فى هذا الموضع ؟

٨ – ٢٣ كيف يمكن إنشاء ملف بيانات عشوائى ؟

٨ – ٢٤ كيف يمكن كتابة برنامج حيث يوصف اسم ملف البيانات أثناء وقت التشغيل ؟ وما هى جمل التعامل مع البيانات التى يجب استخدامها من أجل إنجاز ذلك ؟

٨ – ٢٥ عند توصيف ملف البيانات العشوائى أثناء وقت التشغيل ، هل يجب أن نضمن الملحق فى اسم الملف ؟

٨ – ٢٦ لخص الفرض من كل من جمل التعامل مع الملفات التالية : FILES و FILE و INPUT و READ و PRINT و WRITE و IF END و QUOTE و SCRATCH و SET

٨ – ٢٧ كيف تختلف جمل التعامل مع الملفات INPUT و PRINT عن READ و WRITE ؟

٨ – ٢٨ كيف تختلف جمل التعامل مع الملف FILES و FILE عن بعضها ؟

٨ – ٢٩ ما هو الفرض من الدوال المكتبية LOC و LOF ؟

مسائل محلولة
Solved Problems

٨ – ٣٠ عدة جمل بيسك أو مجموعات من الجمل ، مبينة فيما يلى ، بعض منها مكتوب بصورة غير صحيحة . تعرف على كل الأخطاء .

(١)

```
10 FILES DATAOLD,DATANEW
```

اسم الملف لا يمكن أن يتعدى الحروف السته فى معظم أنظمة بيسك .

(ب)
```
50 PRINT #3,N,N$,P+Q,LOG(T)
```

صحيحة ، إذا تم تحديد قناة البيانات 3 لملف البيانات المتسلسل

(ج)
```
25 READ :2,N$,M$,C1,C2
```

لا يمكن أن يحتوى ملف عشوائى على كل من الثوابت الرقية والحرفية معا .

(د)
```
150 IF END #1, THEN STOP
```

يجب أن يستعاض عن كلمة STOP برقم جملة .

(هـ)
```
10 FILES DATA1%,DATA2%
   ...
40 READ :1,C1,C2
   ...
80 WRITE :2,C1,C2
```

صحيحة .

(و)
```
10 FILES SALES
   ...
75 SET #1,P
80 INPUT #1,A,B,T$,G
```

تستخدم جملة SET مع ملفات البيانات العشوائية فقط .

(ز)
```
60 IF P=LOF(2) THEN 175
```

صحيحة إذا تم تحديد قناة البيانات 2 لملف البيانات العشوائى .

٨ – ٣١ اكتب جملة أو أكثر من جمل بيسك أو أوامر النظام لكل من المواقف الموصوفة فيما يلى :

(ا) أنشئ ملف بيانات متسلسل SALES احتفظ بالملف ثم أصدر قائمة به بعد أن يتم إدخاله .

```
NEW OR OLD--> NEW
NEW FILE NAME--> SALES
 10 ...  }
 20 ...  }
 ...     }  Data file SALES
200 ...  }
SAVE
LIST
```

(ب) حدد قناة البيانات 1 لملف البيانات المتسلسل SALES

```
10 FILES SALES
```

(جـ) حدد قناة البيانات 1 لملف البيانات المتسلسل الممثل بالمتغير F$

```
10 INPUT F$
20 FILE #1,F$
```

(د) يتكون كل سطر من ملف البيانات المتسلسل FILE 1 من رقم السطر ، تتبعه قيم المتغيرات A و B و P$ و Q$ ونرغب فى إيجاد قيمة C لكل قيمة من قيم A و B حيث :

C=SQR(A*B)

تمت كتابة القيم A و B و C و P$ و Q$ على ملف بيانات متسلسل آخر يسمى FILES بفرض أن مجاميع البيانات تقرأ من القناة رقم 2 وتكتب على القناة رقم 4 .

```
10 FILES,FILE1,,FILE2
20 QUOTE #4
30 SCRATCH #4
40 INPUT #2,N,A,B,P$,Q$
50 LET C=SQR(A*B)
60 PRINT #4,N,A,B,C,P$,Q$
70 IF END #2, THEN 90
80 GO TO 40
90 END
```

(هـ) يقرأ برنامج بيانات من ملف البيانات المتسلسل MASTER ثم تكتب البيانات المعدلة على ملف البيانات المتسلسل REVISE . وفى وقت لاحق ترغب فى استخدام هذا البرنامج لقراءة البيانات من الملف REVISE وحيث أن البرنامج لن يتغير ، فإننا نرغب فى تغيير الاسم REVISE (ملف الخرج الجديد) إلى MASTER (ملف الإدخال القديم) . بين كيف يمكن إنجاز ذلك .

```
OLD
OLD FILE NAME-->MASTER
SCRATCH
OLD
OLD FILE NAME-->REVISE
RENAME MASTER
```

(و) سوف يكتب برنامج بيانات على ملف عشوائى MASTER . بين كيف يتم تعريف ملف خال يسمى MASTER ثم الاحتفاظ به لتشغيل البرنامج .

```
NEW
NEW FILE NAME-->MASTER
SAVE
```

(ز) حدد الموقع الحالى لمؤشر ملف بيانات عشوائى على القناة 3 .حول التحكم إلى نهاية البرنامج إذا وضع المؤشر عند آخر مكان فى ملف البيانات .

```
100 LET P=LOC(3)
110 IF P=LOF(3) THEN 250
    ...
250 END
```

(ح) اقرأ كمية صحيحة موجبة من الكونسول . ثم ضع المؤشر الخاص بقناة البيانات رقم 5 إلى المكان المشار إليه بواسطة الكمية المدخلة .

```
510 INPUT P5
160 SET :5,P5
```

مسائل تكميلية

Supplementary Problems

٨ – ٣٢ قرر أيا من جمل التعامل مع الملف التالية متاحاً على الجهاز الخاص بك : FILES و FILE و INPUT و READ و PRINT و WRITE و IF END و QUOTE و SCRATCH و SET (أو RESET) و RESTORE و MARGIN و PAGE . هل توجد أي جمل للتعامل مع الملف متاحة علاوة على ذلك ؟

٨ – ٣٣ راجع الغرض من كل من جمل التعامل مع الملف المتاحة على الجهاز الخاص بك . لخص القواعد النحوية لكتابة كل جملة .

٨ – ٣٤ عدة جمل بيسك أو مجموعات من الجمل ، مبينة فيما يلي . بعضها مكتوب بطريقة غير صحيحة . تعرف على كل الأخطاء .

(*a*) 35 READ #1,N,A,B,C,P$,Q$

(*b*) 160 WRITE :2,X,Y,X+Y,X−Y,P$

(*c*) 80 SET :2,LOC(1)+2
90 WRITE :2,X1,X2,X3

(*d*) 10 FILES NAMES$20,ACCTS%

(*e*) 10 FILES MASTER
20 SCRATCH MASTER
30 QUOTE MASTER

(*f*) 10 INPUT F$
20 FILES F$

(*g*) 100 SET :1,LOF(1)+3
110 READ :1,L,M,N

(*h*) 10 FILES NAMES$20,ACCTS%
. . .
75 SET :1,P1
80 READ :1,N$
85 SET :2,P2
90 WRITE :2,N$

٨ – ٣٥ اكتب جملة أو أكثر من جمل بيسك أو أوامر النظام لكل من المواقف الموصوفة فيما يلي :

(أ) حدد قنوات البيانات 1 و 3 لملفات البيانات المتسلسلة LIST 1 و LIST 2 على الترتيب .

(ب) حدد قناة البيانات 1 لملف البيانات العشوائي الحرفي المسمى NAMES وحدد قناة البيانات 2 لملف البيانات العشوائي الرقمي المسمى ACCTS . افرض أن كل سلسلة الحروف في NAMES سوف تتكون من 25 حرفاً أو أقل .

(جـ) حدد قنوات البيانات 2 و 5 لكل من ملفات البيانات العشوائية المشلة بالمتغيرات F$ و G$ على الترتيب .

(د) أنشيء ملف البيانات المتسلسل TAPES احتفظ بالملف ثم اصدر قائمة بمحتوياته بعد الانتهاء من إدخال جميع البيانات.

(هـ) سوف يكتب برنامج البيانات على ملف بيانات عشوائي ITEMS . بين كيف يمكن لملف بيانات خال يسمى ITEMS أن يعرف ويحفظ استعداداً لتشغيل البرنامج .

(و) يقرأ برنامج البيانات من ملفات بيانات متتالية OLD 1 و OLD2 ثم يكتب البيانات المعدلة على الملفات المتتالية NEW 1 و NEW2 وفي وقت لاحق يمكن أن نطلب استخدام نفس البرنامج لقراءة ملف البيانات NEW 1 و NEW 2 . كيف يمكن تغيير الأسماء السابقة (NEW 1 و NEW 2) حتى يمكن استخدامها كملفات إدخال ؟

(ز) كيف يمكن إعادة كتابة البرنامج الموصوف فى الجزء (و) حتى لانحتاج إلى إعادة تسمية ملفات الإخراج قبل قراءتها ؟ إفرض أن ملفات الإدخال سوف تحدد بقنوات البيانات 1 و 2 وملفات الإخراج بالقنوات 3 و 4 .

(ح) يتكون كل سطر فى ملف البيانات المتسلسل المحدد بقناة البيانات 5 من رقم السطر ، يتبعه قيم المتغيرات F$ و X و Y و Z و G$. إفرض أننا نرغب فى كتابة قيم Z و F$ و G$ على ملف بيانات متسلسل محدد بقناة البيانات 3 . بين كيف يمكن إنجاز ذلك .

(ط) إنسخ ملف البيانات العشوائى الرقمى المحدد بقناة البيانات 5 إلى الملف المحدد بقناة الاتصال 3 .

(ى) إقرأ كية صحيحة موجبة من الكونسول . حدد موضع المؤشر لقناة البيانات 6 إلى المكان المشار إليه بواسطة الكية المدخلة . إقرأ قيمة X من هذا الموضع ، ثم اكتب قيمة X فى الموضع المناظر على قناة البيانات 2 .

(ك) حدد موضع مؤشر ملفات البيانات العشوائية المحددة بقنوات البيانات 1 و 4 حول التحكم إلى الجملة رقم 200 إذا كان كل من المؤشرين لها نفس القيمة (أى تشير إلى نفس المواضع بالترتيب) . وإلا اجعل قيمة مؤشر قناة البيانات 2 إلى القيمة الأكبر من القيمتين .

(ل) حدد المواضع الأخيرة من ملفات البيانات العشوائية التى تحددها قنوات البيانات 5,3 . حول التحكم إلى الجملة رقم 25 إذا كانت المواضع الأخيرة مختلفة عن بعضها .

مسائل للبرمجة
Programming Problems

٨ – ٣٦ عدل البرنامج المعطى فى مثال ٨ – ٤ بحيث يمكن إدخال أسماء الملفات من الكونسول أثناء تنفيذ البرنامج . أيضاً ، ضمن اختياراً يسبب تخزين مجموعات من البيانات بترتيب متوسطات الفصل الدراسى وليس بترتيب أسماء الطلبة هجائياً .

٨ – ٣٧ عدل البرنامج المعطى فى مثال ٨ – ١٠ حيث يمكن عمل أى من الاختبارات التالية :

(أ) ببساطة اطبع مستوى المخزون لأى رقم مخزون .

(ب) عرف « كتلة » من أرقام المخزون بقراءة أول وآخر رقم مخزون فى الكتلة . اطبع مستوى المخزون لكل رقم مخزون فى الكتلة .

(حـ) اطبع مستوى المخزون لكل رقم مخزون فى ملف البيانات .

(د) اطبع أرقام المخزون ومستويات المخزون المناظرة لكل البنود التى مستوى مخزونها أقل من مستوى مخزون محدد .

(هـ) اطبع أرقام المخزون ومستويات المخزون المناظرة لكل البنود التى مستوى مخزونها أكبر من مستوى مخزون محدد .

٨ – ٣٨ عدل البرنامج المعطى فى مثال ٨ – ١٣ حتى يمكن القيام بأى من الاختبارات التالية :

(أ) إظهار قائمة كاملة لملف البيانات على الكونسول .

(ب) طباعة كل سلاسل الحروف مبتدئاً من حرف معين وذلك على الكونسول .

(حـ) طباعة كل سلاسل الحروف التى تسبق سلسلة حرفيه معينة .

(د) طباعة كل سلاسل الحروف الموجودة وراء نطاق سلسلة حرفية معينة .

٨ – ٣٩ أعد كتابة البرنامج الخاص بكل من المسائل التى تظهر قائمتها فيما يلى حيث يمكن أن تقرأ البيانات أو تكتب البيانات على ملف . (لاحظ : أن بعض المسائل سوف تتطلب قراءة البيانات من ملف بيانات وطباعة النتائج المحسوبة على الكونسول وهناك مسائل أخرى تقبل المدخلات من الكونسول ، وكتابة المخرجات على ملف البيانات . وفى حالات قليلة يمكن أن يكون المطلوب هو قراءة البيانات المدخلة من ملف بيانات ثم كتابة المخرجات على ملف بيانات آخر) .

(أ) مثال ٤ – ١٦
(ب) مثال ٤ – ١٨
(حـ) مسألة ٤ – ٤٨ (هـ)
(د) مثال ٥ – ٥
(هـ) مسألة ٥ – ٤٨
(و) مسألة ٥ – ٤٩
(ز) مسألة ٥ – ٥٠
(ح) مسألة ٥ – ٥١
(ط) مسألة ٥ – ٥٢
(ى) مسألة ٥ – ٥٣
(ك) مسألة ٥ – ٥٤
(ل) مسألة ٥ – ٥٥
(م) مسألة ٥ – ٥٦
(ن) مسألة ٥ – ٥٧ (أ)
(ش) مسألة ٥ – ٥٧ (ب)
(س) مسألة ٥ – ٥٧ (حـ)
(ص) مسألة ٥ – ٥٧ (د)
(ض) مسألة ٥ – ٥٧ (ح)
(ط) مثال ٦ – ٢٦
(ظ) مثال ٦ – ٢٨
(ع) مسألة ٦ – ٤٨
(غ) مسألة ٦ – ٥٢ (هـ)
(ف) مسألة ٦ – ٥٢ (ى)

٨ – ٤٠ اكتب برنامج بيسك كاملا ينشأ وينتفع من ملف بيانات متسلسل يحتوى على أسماء وعناوين وأرقام تليفونات . ضمن احتياط لكل من الخصائص التالية :

(أ) إضافة سجل جديد للملف (أى اسم جديد وعنوان ورقم تليفون) .

(ب) أوجد سجلا معيناً ثم اعرضه على النهاية الطرفية وذلك مبنى على أساس التعرف على المعلومات التى تدخلها من الكونسول (مثال ، إسم أو عنوان أو رقم تليفون) .

(حـ) احذف سجلا معيناً ، مبنياً على أساس التعرف على المعلومات التى تدخل من الكونسول .

(د) رتب السجلات أبجدياً ، وذلك مبنى على أساس آخر اسم فى كل سجل .

(هـ) أعط قائمة (أى اطبع) الملف بالكامل .

(و) أوقف الحسابات .

٨ – ٤١ كرر المسألة ٨ – ٤٠ مستخدماً ملف بيانات عشوائياً . ثم قارن نسخة ملف البيانات المتسلسل من وجهة نظر سهولة البرمجة وسرعة التنفيذ .

الجزء الثالث : بيسك الحاسب الدقيق

الفصل ٩

التحسينات على البيسك
Enhancements to BASIC

من المفترض أن كل الحاسبات الدقيقة تدعم البيسك كلغة برمجة ابتدائية . وتحتوى بعض الحاسبات الدقيقة على مفسر البيسك داخل ذاكرة للقراءة فقط (ROM) بحيث تصبح اللغة فى الحقيقة جزءًا من دائرة الحاسب الداخلية . وتقرأ أجهزة أخرى مفسر البيسك ليدخل ذاكرة الحاسب من جهاز تخزين كبير (مثل قرص مرن) عندما نحتاجها . وفى كل من الحالتين ، فإن البيسك يطوع نفسه طبيعياً لبيئة الحاسب الدقيق . وفى الحقيقة فإن إتاحة البيسك أصبح من العوامل المعنوية جداً فى التطوير التجارى لسوق الحاسب الدقيق .

ومعظم الحاسبات الدقيقة تدعم نسخ بيسك معقدة وتتضمن تحسينات كثيرة غير موجودة فى النسخ التقليدية من اللغة . هذا بالإضافة إلى أن هذه الصور المعقدة من البيسك فى معظمها هى أنواع مختلفة من ميكروسوفت بيسك (أعدتها مؤسسة Microsoft Corporation) . وعلى سبيل المثال فإن الحاسبات الدقيقة التى قامت بتسويقها شركات TEXAS INSTRUMENTS, ZENITH AND IBM AT& T, APPLE-RADIO SHACK) من بين مجموعة أخرى كلها تتضمن بعض الاختلافات فى ميكروسوفت بيسك . ولهذا فإن المادة التى سنعرضها فى الفصول الأربعة التالية من هذا الكتاب ستكون مبنية على أساس الخصائص الموجودة فى ميكروسوفت بيسك (على الأخص نسخة ميكروسوفت بيسك على حاسبات IBM الشخصية الشائعة) .

وتهدف المادة المعروضة فى هذا المرجع إلى تمثيل هذه الخصائص المتاحة على حساب دقيق نموذجى . ومع كل فعلى القارىء أن يفهم أن كل هذه الخصائص قد لا تكون منفذة على كل حاسب . هذا بالإضافة إلى أن هذه الخصائص المتاحة عموما قد يتم تنفيذها بشكل مختلف نوعاً ما على كل حاسب . وعلى ذلك ، فإن هذه المادة لابد من اعتبارها كنظرة عامة على بيسك الحاسب الدقيق بدلاً من اعتبارها وثيقة مرجعية دقيقة . وعلى القارىء إذا ما أراد معلومات خاصة حول نسخة معينة من اللغة أن يرجع إلى الكتيب المرجعى المناسب للمبرمج .

وسنجد فى ملحق (د) من هذا الكتاب تلخيصاً لأكثر الخصائص الشائعة لميكروسوفت بيسك .

٩ــ١ توسعات أولية فى اللغة ELEMENTARY LANGUAGE EXTENSIONS

معظم النسخ المختلفة لبيسك الحاسب الدقيق تسمح عموماً بمدى أوسع فى تعريف واستخدام المتغيرات والمعاملات عن زميلاتها التقليدية . بعض التوسعات الأكثر شيوعاً تم مناقشتها أدناه .

أسماء متغيرات أكبر Larger Variable Names

تسمح معظم نسخ بيسك الحاسب الدقيق بأسماء المتغيرات الأطول من حرفين . وفى الحقيقة ، فإن بعض نسخ اللغة لا تضع أى قيود على أقصى عدد مسموح به من الحروف . وهذه خاصية ملائمة جداً ، حيث أنها تسمح لأسماء المتغيرات بأن تناظر البنود التى تمثلها . ومع كل ، فعادة ما تكون الحروف العديدة الأولى معنوية فى التعرف على المتغير . ويمكن أن يتراوح العدد الشائع للحروف المعنوية من 2 إلى 40 حرفاً . وقد يصل عدد الحروف لأسماء المتغيرات على حاسب IBM الشخصى إلى 40 حرفا ، وكلها معنوية .

مثال ٩ - ١

فيما يلى عدة أسماء متغيرات :

NAME	AREA
ADDRESS	SIZE
PATIENT	TABLE
PAYROLL	VECTOR
TAX	VELOCITY

سوف تتعرف كثير من نسخ البيسك للحاسب الدقيق على كل من هذه الأسماء كمتغيرات منفصلة . ومع كل فإذا كان الحرفان الأولان فقط معنويين فإن المتغيرات TAX, TABLE, PAYROLL, PATIENT, VELOCITY, VECTOR لا يمكن تمييز بعضها عن بعض .

ويجب تجنب استخدام الكلمات الدالة فى البيسك كأسماء متغيرات مثل (PRINT و DATA و NEXT الخ) لأنها سوف تسبب تشويشاً للمترجم ، وغالباً ماينتج عنها رسالة خطأ . وعموماً ، فيجب على المبرمج أن يحرص على عدم استخدام أى من الكلمات الدالة الأقل شيوعاً كأسماء متغيرات (مثل NAME, CHAIN, SWAP الخ) .

الأنواع المتعددة للبيانات الرقمية Multiple Numeric Data Types

تميز معظم نسخ البيسك للحاسب الدقيق بين الكميات الصحيحة والكميات الحقيقية والكميات المزدوجة الدقة .

والكميات الصحيحة هى الأرقام الصحيحة الكاملة سواء أكانت موجبة أم سالبة والتى تقع فى المدى مابين 32,768 — و 32,767 ومثل هذه الكميات لاتتعرض لأخطاء التقريب . من ثم فهى مفيدة كعدادات أو ادلة لعناصر المجموعات المتراصة أو كمتغيرات جارية (مؤشرات) فى الحلقات التكرارية FOR-TO أو جمل ON GO TO و .. الخ . يكون من المستحب أيضاً استخدام الكميات الصحيحة عند القيام باختبارات منطقية معينة باستخدام جملة IF-THEN .

والكميات الحقيقية هى قيم رقية تمت مناقشتها فى الفصل الثانى . والكمية الحقيقية يمكن أن تتضمن أولا تتضمن مكونات كسرية (عشرية) و/ أو أس ، وقد يكون مقدار الكمية الحقيقية صغيراً أى 2.9E-39 ، وكبيراً إلى 1.7E + 38 (والقيمة 0.0 هى أيضاً قيمة حقيقية مقبولة) .

وفى داخل ذاكرة الحاسب يتم تمثيل الكميات الحقيقية والكميات الصحيحة بطريقة مختلفة حتى إذا لم يكن هناك مركبة كسرية . وتكون الكميات الحقيقية غير دقيقة (تكون معرضة لأخطاء التقريب) بينما تكون الكميات الصحيحة مضبوطة .

والكميات المزدوجة الدقة هى أساساً كميات حقيقية (مفردة الدقة) ولكن لها عدد أكبر من الأرقام المعنوية . والكمية المتضاعفة الدقة النموذجية تمثل بدلالة 16 رقماً معنوياً ، بينما الكمية الحقيقية سوف تحتوى فقط على 6 أو 7 أرقام معنوية . وأيضاً يستخدم الحرف D لتمثيل الأس بدلا من الحرف E .

وتتعرف العديد من نسخ بيسك الحاسبات الدقيقة على أنواع أخرى من البيانات الرقمية مثل الثمانى (الأساس 8) والسادس عشر (الأساس 16) ، ومع كل فاستخدام أنواع البيانات هذه يخرج عن نطاق مناقشتنا الحالية .

مثال ٩ – ٢

عدة أنواع مختلفة من الأرقام مبينة فيما يلى :

النوع	الرقم
صحيحة أو حقيقية	16458
حقيقى	36.55
حقيقى	−0.666667E−3
مزدوج الدقة	−0.6666666666666667D−3

عندما نقوم بتمثيل رقم بمتغير رقمى فلابد أن يكون الرقم والمتغير من نفس النوع وبذلك فإن الكميات الصحيحة والحقيقية والمزدوجة الدقة لابد أن تمثل بمتغيرات صحيحة وحقيقية مزدوجة الدقة على الترتيب . وعموماً فإن الأنواع المختلفة من المتغيرات تعرف بواسطة آخر حرف من اسم المتغير . وكنموذج فإن المتغير الصحيح ينتهى دائماً بعلامة (%) ، وينتهى المتغير الحقيقى بعلامة التعجب (!) ، والمتغير المزدوج الدقة ينتهى بعلامة الجنيه (#) . (تذكر أن المتغير الحرفى ينتهى دائماً بعلامة الدولار ($)) .إن لم ينته اسم المتغير بأى حرف مميز ، فسوف يترجم على أنه متغير حقيقى .

ويطلق على هذه النهايات الخاصة أحياناً ملاحق . والأنواع العديدة من الملحقات يمكن تلخيصها فيما يلى :

المتغير	الملحق
صحيح	%
حقيقى (أو دقة مفردة)	! (أو بدون ملحق)
متضاعف الدقة	#
حرفى	$

مثال ٩ – ٣

عدة متغيرات بيسك وأنواع البيانات المناظرة لها مبينة فيما يلى :

المتغير	نوع البيانات
COUNT%	صحيحة
NAME$	حرفيـة
PAY	حقيقية (مفردة الدقة)
TAX !	حقيقية (مفردة الدقة)
ERROR #	متضاعف الدقة

وفى حالة نفس الاسم وبملاحق مختلفة سوف يترجم على أنه متغير مختلف .

مثال ٩ – ٤

نفرض أن المتغيرات

A و A$ و A% و A #

كلها تظهر فى نفس برنامج البيسك . وسوف تترجم كل منها كمتغير مستقل ومنفصل حيث أن لها ملاحق مختلفة ، ومن ثم فإنها تمثل أنواع بيانات مختلفة .

ويجب أن يعرف المبرمج أن متطلبات الذاكرة لأنواع البيانات المختلفة ليست متماثلة ، فتتطلب الكميات الصحيحة أقل مساحة من الذاكرة 2 bytes ، بينما تتطلب الكميات المزدوجة الدقة أكبر مساحة من الذاكرة 8 bytes . وأيضاً ، فإن أسماء المتغيرات الطويلة تتطلب ذاكرة أكبر من التي تتطلبها أسماء المتغيرات القصيرة تماماً كما تتطلب السلاسل الحروف الطويلة ذاكرة أكبر من السلاسل الحروف القصيرة . ويجب أن تؤخذ هذه العوامل فى الاعتبار عند استخدام الحاسبات الدقيقة حيث أن كمية الذاكرة المتاحة يمكن أن تكون قليلة .

ويجب تجميع مثل هذه الاعتبارات لوقت التنفيذ . وكقاعدة ، فإن البرامج التى تستخدم متغيرات صحيحة يتم تشغيلها أسرع من البرامج التى تحتوى على متغيرات حقيقية أو مزدوجة الدقة . وبذلك فإن البرنامج الذى يتضمن حسابات رقمية بكمية كبيرة يجب أن ينتفع بالمتغيرات الصحيحة كلما كان ذلك عملياً .

العمليات الحسابية المختلطة Mixed Arithmetic Operations

عند تنفيذ عمليات حسابية بها بيانات رقمية مختلفة النوع يتم التعبير عن نتائجها بأعلى مستوى من الدقة . وعلى ذلك فالعمليات الحسابية التى تشتمل على بيانات صحيحة وحقيقية تكون نتائجها حقيقية ، وبالمثل فالعمليات الحسابية التى تستخدم إما بيانات صحيحة ومزدوجة الدقة أو بيانات حقيقية ومزدوجة الدقة سوف تكون نتائجها مزدوجة الدقة .

مثال ٩ – ٥

فى كل من التعبيرات التالية نفرض أن I% = 4, R! = −0.2 and D# = 0.16666666D−4

التعبير	القيمة
R! + 5	حقيقية 4.8
I% * R!	حقيقية −0.8
3 * I% * D#	مزدوجة الدقة 0.199999992D−3
(1 + R!) * D#	مزدوجة الدقة 0.133333328D−5

والآن ادرس جملة التحديد الرقية (أى جملة LET) حيث المتغير الموجود على اليسار والكمية الموجودة على اليمين من أنواع مختلفة . سوف تحول الكمية الموجودة فى اليمين أتوماتيكياً لنفس نوع للتغير الموجود فى اليسار . لاحظ أن ذلك سوف يتسبب فى تقريب الكمية الكسرية الموجودة فى اليمين إذا ظهر متغير صحيح فى اليسار (بعض نسخ البيسك سوف تقوم بالبتر بدلا من التقريب للكمية الكسرية) .

مثال ٩ – ٦

فى كل من جمل التحديد التالية ، نفرض أن I% = 4 و R! = −0.2 و D# = 1.6666666 D-4 .

جملة التحديد	النتيجة
10 LET N% = 3 * R!	N% = −1
20 LET FRACT = 1/3	FRACT = 0.3333333
30 LET FRACT# = 1/3	FRACT# = 0.3333333333333333
40 LET ANS% = I%↑2/3	ANS% = 5
50 LET RATIO = (I% − R!)/D#	RATIO = 2.52E+5

(لاحظ التقريب الذى نتج عند تنفيذ الجملة رقم 10 . كان البتر سيؤدى إلى N% = 0 بدلا من N% = −1) .

تذكر أن الكمية الرقية لايمكن أن تستخدم لتحديد متغير حرفى ، وبالعكس . وتذكر أيضاً أن الكلمة المرشدة LET اختيارية فى عديد من نسخ البيسك للحاسبات الدقيقة .

معاملات إضافية Additional Operators

تتضمن بعض نسخ البيسك معاملين حسابين إضافيين : وهما القسمة الصحيحة (\)، والكية المتبقية الصحيحة (MOD) . فى القسمة الصحيحة يقرب كل من الرقين المعطيين أولا لرقم صحيح ، ثم تنفذ عملية القسمة ، ثم يبتّر خارج القسمة بعد ذلك . تمدنا عملية إيجاد الكية المتبقية الصحيحة بالكية المتبقية بعد تنفيذ عملية القسمة الصحيحة .

مثال ٩ — ٧

تمثل فيما يلى عدة عمليات للقسمة الصحيحة وإيجاد الباقى الصحيح :

13\5 = 2	13 MOD 5 = 3
8.6\2.7 = 3	8.6 MOD 2.7 = 0
8.3\2.7 = 2	8.3 MOD 2.7 = 2
8.3\2.2 = 4	8.3 MOD 2.2 = 0

بعض النسخ الأكثر بساطة من بيسك الحاسبات الدقيقة تتعرف فقط على البيانات الرقية من النوع الصحيح . فى مثل هذه الحالات فإن معامل القسمة العادى (\) سوف يعنى عملية القسمة الصحيحة (مع بتّر خارج القسمة) .

وتتضمن بعض نسخ البيسك للحاسبات الدقيقة المعاملات المنطقية مثل AND و OR و NOT . أو معاملين (AND و OR) تسمح بدمج شرطين أو أكثر من شروط « صحيحة خاطئة » . ويستخدم المعامل الثالث (NOT) لنفى شرط « صحيح — خطأ » (أى يغير « الصحيح » إلى « خطأ » أو « الخطأ » إلى « صحيح » (تسمح هذه العمليات بتضمين شروط أكثر تعقيداً فى جملة IF—THEN .

مثال ٩ — ٨

مبين فيما يلى عدة جمل IF–THEN تستخدم الشروط المنطقية المعقدة .

(أ)

```
10 IF (X<10) AND (Y>0) THEN 90
20 PRINT X,Y
```

سوف تنفذ الجملة رقم 90 مباشرة إذا كانت قيمة X أقل من 10 وقيمة Y أكبر من الصفر ، وإلا فسوف تنفذ الجملة رقم 20 مباشرة .

(ب)

```
50 IF (COUNT>99) OR (N$ = "END") THEN 175
60 GO TO 30
```

سوف تنفذ الجملة رقم 175 مباشرة إذا تعدت قيمة COUNT الرقم 99 أو إذا كانت قيمة N$ هى "END" (أو كليهما) . أما إذا كان كلا الشرطين غير صحيح فسوف تنفذ الجملة رقم 60 مباشرة .

(ﺟ)

```
75 IF NOT ((X<10) AND (Y>0)) THEN 200
80 PRINT X,Y
```

هذا المثال عكس المثال (أ) . فى الحالة الراهنة سوف تنفذ الجملة رقم 200 إذا كان الشرط :

```
(X<10) AND (Y>0)
```

غير صحيح ، أى إذا كانت قيمة X أكبر أو تساوى 10 أو إذا كانت قيمة Y أقل من أو تساوى 0 ، وإلا فسوف تنفذ الجملة رقم 50 مباشرة . لاحظ أن جملة IF كان يمكن كتابتها أيضاً :

```
75 IF (X>= 10) OR (Y<= 0) THEN 200
```

و كتطبيق عملى فإن معامل NOT يلقى استخداماً أقل نسبياً فى برامج البيسك الحقيقية .

وتحتوى بعض نسخ البيسك للحاسبات الدقيقة على معاملات إضافية منطقية XOR (Exclusive OR) ، EQV (Equivalent) و IMB (Implies) وباختصار عندما تستخدم لتوصيل عاملين منطقيين، فإن XOR تنتج فى حالة صحيحة إذا كان أحد العوامل صحيح والآخر غير صحيح ، وتنتج عن EQV شرط إذا كان كل من العاملين له نفس القيمة المنطقية (إما صحيحين أو غير صحيحين) أو ينتج عن IMB حالة صحيحة إذا كان كل من العاملين أو إذا كان العامل الأول غير صحيح (بصرف النظر عن قيمة العامل الثانى) . وحيث أن هذه العوامل المنطقية الثلاثة لا تستخدم إلا قليلاً فإننا سنتجنب مناقشتها فى هذا النص .

والتدرج الهرمى النموذجى الكامل للمعاملات الحسابية والمنطقية ومعاملات الترابط هو :

١ - الرفع للأس	↑ أو
٢ - النفى (أى ، أن يسبق المتغير علامة الطرح)	
٣ - الضرب والقسمة	* /
٤ - القسمة الصحيحة	\
٥ - البـــاقى الصحيح .	MOD
٦ - الجمع والطرح	+ -
٧ - عمليات الترابط	= <> <= < >= >
٨ - NOT المنطقية	NOT
٩ - AND المنطقية	AND
١٠ - OR المنطقية	OR
١١ - XOR المنطقية	XOR
١٢ - EQV المنطقية	EQV
١٣ - IMP المنطقية	IMP

ويمكن أن يتغير التسلسل الهرمى من نسخة بيسك لأخرى . وبداخل مجموعة التسلسل الهرمى تنفذ العمليات من اليسار إلى اليمين .

عدة نسخ معدلة من البيسك يمكن أن تتضمن أيضاً معامل الوصل وهو يستخدم فى دمج السلاسل الحرفية . وغالباً مايمثل هذا المعامل بواسطة علامة الجمع (+) ، بالرغم من أن علامة & أو الفصلة (,) يمكن أن تستخدم لنفس الغرض فى نسخ معينة من اللغة .

مثال ٩ - ٩

نفرض أن المتغيرات الحرفية $ A و $ B تمثل بالسلاسل الحرفية "MICRO" و "COMPUTER" على الترتيب . وبذلك فإن الجملة

```
10 PRINT A$ + B$
```

سوف تسبب طباعة سلسلة حرفية (واحدة) وهى :

```
MICROCOMPUTER
```

أن تطبع . (بفرض أن رمز الوصل هو علامة الجمع « + ») .

وبالمثل .. فإن الجملة

20 PRINT "SEVEN" + "TEEN"

سوف يترتب عليها ظهور السلسلة الحرفية الواحدة :

SEVENTEEN

جمل متعددة فى السطر Multiple Statements per Line

تسمح معظم نسخ بيسك المعدلة بظهور جمل متعددة فى نفس السطر . وعادة ما تكون علامة الوقف الاستدراكى (:) هى التى تستخدم فى فصل إحدى هذه الجمل عن الأخرى .

مثال ٩ – ١٠

عدة سطور تحتوى على جمل متعددة مبينة فيما يلى :

```
10 LET A=0.25 : B=0.5 : C=-0.125
20 PRINT "X="; : INPUT X
30 FOR I%=1 TO N% : READ A(I%) : NEXT I%
```

واستخدام هذه الخاصية يلائم غرضين : أولا أنه يسمح لمجموعة متتالية من الجمل المترابطة أن تظهر فى صورة مجموعة منطقية ، وثانياً ، أنه يساهم فى استخدام الذاكرة بطريقة أكثر فاعلية . وعادة استخدام السطور العديدة الجمل فى برامج يمكن أن تترتب عليه صعوبة نسبية فى قراءة وتصحيح البرنامج .

مثال ٩ – ١١ : توليد أرقام فيبوناسى والبحث عن الأرقام الأولية على الحاسب الدقيق .

Generating Fibonacci Numbers and Searching for Primes on a Microcomputer

رأينا فى المثال ٤ – ١٨ برنامج بيسك كاملاً يولد مجموعة متتالية من أرقام فيبوناسى والتعرف على الأرقام الأولية من بينها . ويبين شكل ٩–١ برنامج بيسك آخر مكتوباً بميكروسوفت بيسك الذى يسمح بالقيام بنفس الحسابات على حاسب IBM الشخصى . يستخدم هذا البرنامج توسعات اللغة العديدة التى تم عرضها سابقاً فى هذا الفصل . وعلى الأخص ، فإننا نرى استخدام أسماء المتغيرات الطويلة ، والمتغيرات من النوع الصحيح ، والجمل المتعددة فى عدة سطور ، وأيضاً استخدام المنطق للتعرف على الأرقام الأولية ، حيث أنه من الممكن استخدام عملية باقى القسمة الصحيحة (MOD) . (قارن السطور 110 إلى 120 فى البرنامج الحالى مع السطور 140 إلى 160 فى البرنامج الأصلى).

```
10 REM ***** GENERATION OF FIBONACCI NUMBERS AND SEARCH FOR PRIMES *****
20 PRINT "Generation of Fibonacci Numbers and Search for Primes":PRINT
30 PRINT "How many Fibonacci numbers";:INPUT N%:PRINT
40 F1%=1:F2%=1
50 PRINT "I=";1,"F=";F1%;" (Prime)"
60 PRINT "I=";2,"F=";F2%;" (Prime)"
70 FOR INDEX%=3 TO N%
80    F%=F1%+F2%:ROOT=SQR(F%):MAX%=ROOT
90    PRINT "I=";INDEX%,"F=";F%;
100   FOR DENOM%=2 TO MAX%
110      REMAINDER%=F% MOD DENOM%
120      IF REMAINDER%=0 THEN 150
130   NEXT DENOM%
140   PRINT " (Prime)";
150   PRINT
160   F2%=F1%:F1%=F%
170 NEXT INDEX%
180 END
```

شكل ٩ – ١

وينتج تنفيذ هذا البرنامج نفس المخرجات المبينة فى شكل ١٠ـ٤ من المثال ٤ـ١٨ . وشكل ٩ـ٢ يبين نموذجاً لمجموعة مخرجات يولدها هذا البرنامج (رغم أن المخرجات عادة يتم عرضها على شاشة T V بدلاً من طباعتها على نسخة ورقية على النهاية الطرفية . واستجابات المستخدم موضوع تحتها خط .

ويجب أن نذكر هنا ، أنه بالنسبة لحاسب الدقيق ذى الأرقام الثنائية الثمانية تكون أكبر قيمة يمكن إعطاؤها للمتغير N% (فى السطور رقم 30) هو 23 ، حيث أن أى قيمة أكبر من 23 سوف يترتب عليها رقم من أرقام فيبوناسى يتعدى 32, 767 وأكبر رقم صحيح مسموح به على معظم الحاسبات الدقيقة.ويمكن لهذا القيد أن يحد من الاستخدام العملى لهذا البرنامج على حاسب دقيق ذى ثمانية أرقام ثنائية.
لاحظ أن السلسلة المبينة فى السطور 20 ,30 ,50 ,60 ,140 تحتوى على كل من الحروف الكبيرة والصغيرة والتى تفرق بينها معظم الحاسبات الدقيقة فى حالة السلاسل فقط .

```
Generation of Fibonacci Numbers and Search for Primes

How many Fibonacci numbers? 23

I= 1          F= 1   (Prime)
I= 2          F= 1   (Prime)
I= 3          F= 2   (Prime)
I= 4          F= 3   (Prime)
I= 5          F= 5   (Prime)
I= 6          F= 8
I= 7          F= 13   (Prime)
I= 8          F= 21
I= 9          F= 34
I= 10         F= 55
I= 11         F= 89   (Prime)
I= 12         F= 144
I= 13         F= 233   (Prime)
I= 14         F= 377
I= 15         F= 610
I= 16         F= 987
I= 17         F= 1597   (Prime)
I= 18         F= 2584
I= 19         F= 4181
I= 20         F= 6765
I= 21         F= 10946
I= 22         F= 17711
I= 23         F= 28657   (Prime)
```

شكل ٩ ـ ٢

٩ ـ ٢ جمل إضافية ADDITIONAL STATEMENTS

تتضمن معظم نسخ البيسك للحاسبات الدقيقة عددا من الجمل المناسبة غير الملائمة وغير المتاحة على نسخ اللغة التقليدية ، العديد من الجمل الإضافية الأكثر شيوعاً سوف تناقش أدناه .

DEFINT, DEFSNG, DEFDBL, and DEFSTR

يكون من المناسب فى بعض الحالات تعريف عدة متغيرات مختلفة لأنواع بيانات معينة . ويمكن إنجاز ذلك مع جمل DEFINT و DEFSING و DEFDBL و DEFSTR (للصحيح ومنفرد الدقة ومزدوج الدقة وبيانات السلاسل الحرفية على الترتيب) بشرط أن كل أسماء المتغيرات لنوع معين تبدأ بنفس الحرف الأول . ومن الممكن أيضاً تعريف مدى للحروف الأولى ، كما هو موضح أدناه .

مثال ٩ – ١٢

برنامج بيسك يحتوى على الجمل التالية :

```
10 DEFINT I
20 DEFDBL X-Z
30 DEFSTR A-C,N
```

تنص الجملة رقم 10 على أن كل المتغيرات المبتدئة بالحرف I سوف تكون متغيرات صحيحة . ويترتب على الجملة رقم 20 أن كل المتغيرات المبتدئة بالحروف X و Y و Z تكون متغيرات مزدوجة الدقة ، وتعرف الجملة رقم 30 أن كل المتغيرات المبتدئة بالحروف A و B و C و N على أنها متغيرات حرفية . (لاحظ أنه لايلزم استخدام الملاحق لأى من هذه المتغيرات)

ويجب أن يفهم أن أسماء المتغيرات التى تتضمن ملحقاً لها أسبقية على أسماء المتغيرات التى تعرف بواسطة جملة نوع – DEF .

IF - THEN

تتضمن معظم نسخ البيسك للحاسبات الدقيقة شكلاً موسعاً من جملة IF-THEN حيث يمكن أن تظهر جملة مستقلة أو أكثر بعد الكلمة الدالة THEN على شرط أن تظهر كلها على نفس السطر . سوف تنفذ هذه الجمل إذا استوفى الشرط المعطى ، وإلا فسوف تنفذ الجملة المبتدئة على السطر التالى مباشرة .

مثال ٩ – ١٣

يحتوى برنامج بيسك على الجمل التالية :

```
50 IF ERROR>0.001 THEN PRINT ERROR
60 LET X = X1
```

سوف تتسبب جملة IF–THEN فى طباعة قيمة ERROR إذا تعدت الكمية 0.001 ، وإلا فسوف يتحول التحكم مباشرة إلى السطر رقم 60 .

مثال ٩ – ١٤

يحتوى برنامج بيسك على الجملة التالية :

```
200 IF FLAG = 1 THEN A = 10 : C$ = "BLUE" : GO TO 35
210 PRINT A
```

إذا كانت FLAG لها القيمة 1 ، فإن المتغيرات A و G$ سوف تعطى القيم 10 و BLUE على الترتيب ، وبعد ذلك يتحول التحكم إلى السطر رقم 35 وإلا ، فسوف ينفذ السطر رقم 210 بعد ذلك (لاحظ أن الكلمة الدالة THEN تتبعها ثلاث جمل منفصلة) .

IF-THEN-ELSE

تسمح بعض نسخ بيسك للحاسبات الدقيقة بتضمين عبارة ELSE فى جملة IF-THEN ، وبذلك فإن الجملة (أو الجمل) التى تتبع الكلمة الدالة THEN سوف تنفذ إذا توفر الشرط المعطى، وإلا فإن الجملة (أو الجمل) التى تتبع ELSE هى التى سوف تنفذ .

مثال ٩ – ١٥

يحتوى برنامج بيسك على جملة IF–THEN–ELSE التالية :

```
80 IF Z < 0 THEN 50 ELSE 120
```

إذا كانت قيمة Z أقل من الصفر ، فإن الجملة الموجودة فى السطر رقم 50 هى التى تنفذ مباشرة ، وإلا فسوف تنفذ الجملة الموجودة فى السطر رقم 120 بعد ذلك .

مثال ٩ – ١٦

والآن إدرس جملة بيسك التالية :

```
80 IF DIFF<0.001 THEN PRINT ANS: GO TO 300: ELSE X=X1: GO TO 35
```

سوف تطبع قيمة ANS ثم يتحول التحكم إلى السطر رقم 300 إذا كانت قيمة DIFF أقل من 0.001 ، وإلا فسوف تحدد قيمة X بالقيمة الحالية للمتغير X1 ثم يتحول التحكم إلى السطر رقم 35 .

وعادة ، يمكن أن تكون جملة IF-THEN-ELSE متداخلة مع جمل IF-THEN-ELSE أخرى ، بالرغم من أن المنطق يمكن أن يكون خادعاً ويتطلب مهارة فائقة ويمكن أن تظهر النتائج مختلفة تماماً عما يمكن أن يتوقعها المبرمج . ويجب أن يرجع القارىء إلى المرجع الخاص به من أجل معلومات أكثر عن جمل IF-THEN-ELSE المتداخلة .

وأخيراً .. يجب أن نذكر أن استخدام جمل IF-THEN-ELSE الممتدة غالباً ماتحسن وضوح المنطق فى برنامج بيسك . ويتم إنجاز ذلك بكتابة التفرعات المشروطة بطريقة مرتبة ومتتالية ، وبذلك ننقص من عدد جمل GO TO غالباً مايطلق على تنظيم البرامج الطريقة **البرمجة الهيكلية** .

ON-GOSUB

جملة ON-GOSUB متاحة فى العديد من نسخ بيسك الحاسبات الدقيقة . وهذه الجملة تشبه جملة ON-GO TO ما عدا أن التحكم يحول إلى برنامج فرعى بدلا من جزء آخر من البرنامج الأصلى . وعند الانتهاء من البرنامج الفرعى ، فسوف يتحول التحكم مرة ثانية إلى الجملة التى تلى جملة ON-GOSUB مباشرة .

مثال ٩ – ١٧

يحتوى برنامج بيسك على الجمل التالية :

```
100 ON FLAG GOSUB 800, 1000, 1200
110 PRINT NAME$
```

يترتب على جملة ON-GOSUB تحويل التحكم إلى أحد ثلاثة برامج فرعية مختلفة ، معتمداً فى ذلك على القيمة المعطاة للمتغير FLAG. فإذا كان FLAG = 1 ، فسوف يتحول التحكم إلى البرنامج الفرعى الذى يبتدىء عند السطر رقم 800 . وبالمثل فإن التحكم سوف يتحول إلى البرنامج الفرعى البادىء فى السطر رقم 1000 إذا كان FLAG = 2 وإلى البرنامج الفرعى البادىء فى السطر رقم 1200 إذا كان FLAG = 3 . لاحظ أن الجملة رقم 110 التى تشتمل على (PRINT NAME $) سوف تنفذ بعد البرنامج دون اعتبار لأى برنامج فرعى تم اختياره .

ON ERROR GO TO

تتعرف معظم نسخ بيسك للحاسبات الدقيقة على عدد من أنواع مختلفة من الأخطاء التى يمكن أن تحدث أثناء تفسير البرنامج أو تشغيله . وجملة ON ERROR GO TO تستخدم لاجراء تصحيح الخطأ بمجرد اكتشافه ، وعلى الأخص فإن جملة ON ERROR GO TO تقوم بتحويل التحكم أوتوماتيكياً لجزء بعيد فى البرنامج عند اكتشاف الخطأ على شرط أن يحدث اكتشاف الخطأ بعد جملة ON ERROR GO TO ويعرف ذلك **بتصيد الخطأ** . وتسمح بتضمين رسائل خطأ أو برامج فرعية لتصحيح الخطأ بداخل البرنامج ، وعادة ما تستخدم جملة RESUME مقترنة بجملة ON ERROR GO TO (انظر ما يلى) .

RESUME

تُستخدم جملة RESUME للإشارة إلى المكان الذى يجب أن نكمل من عنده التنفيذ بعد اكتشاف خطأ ما وبعد تنفيذ البرنامج لتصيد الخطأ .

مثال ٩ — ١٨

مبين فيما يلى جزء لتصيد الأخطاء من برنامج بيسك :

```
10 ON ERROR GO TO 800
   :
50 PRINT "ACCOUNT NUMBER: ";
60 INPUT ACCTNO
   :
800 PRINT "INPUT ERROR—TRY AGAIN"
810 RESUME 50
```

والآن نفرض أننا أدخلنا بيانات إدخال غير صحيحة أثناء تشغيل البرنامج (أدخلنا ـ مثلاً ـ سلسلة حروف بدلاً من رقم فى السطر رقم 60) . وذلك سوف يسبب تفرعاً إلى السطر رقم 800 ، وينتج عنه توليد رسالة خاطئة وسوف يرجع التحكم مرة أخرى للسطر رقم 50 لمحاولة أخرى لقراءة البيانات بطريقة صحيحة .

وسنتحدث بإفاضة حول استخدام جملتى ON ERROR GO TO و RESUME فى الفصل الحادى عشر [انظر قسم ١١ — ٣] .

WHILE and WEND

تتضمن نسخ عديدة من البيسك للحاسبات الدقيقة خاصية تكوين الحلقات التكرارية المشروطة حيث تنفذ مجموعة متتالية من الجمل مراراً ، مادام شرط معين يظل صحيحاً . تستخدم جمل WHILE و WEND لتعريف البداية والنهاية على الترتيب للحلقة التكرارية المشروطة . والشرط الذى يجب أن يستوفى (أى يبقى صحيحاً) يكون متضمناً فى جملة WHILE . وفى داخل الحلقة التكرارية لابد من ايجاد وسيلة لتغيير الشرط وإلا فإن الحلقة ستستمر إلى ما لا نهاية .

مثال ٩ — ١٩

مبين فيما يلى حلقة تكرارية مشروطة وهى تجمع عناصر المصفوفة العددية X ، مادامت قيمة x أكبر من الصفر .

```
10 DIM X(100)
   :
100 LET SUM = 0 : I = 1
110 WHILE X(I)>0
120     SUM = SUM + X(I) : I = I + 1
130 WEND
```

تبدأ الحلقة التكرارية المشروطة فى هذا المثال بالسطر رقم 110 وتنتهى بالسطر رقم 130 . لاحظ استخدام الجمل المتعددة فى السطور 100 و 120. لاحظ أيضاً أن الكلمة الدالة LET قد حذفت من السطر 120 . لابد من ملاحظة أن هذا المثال سوف يولد خطأ إذا كانت كل عناصر الصف موجبة حيث أن الحلقة التكرارية سوف تستمر فى التنفيذ حتى يزيد دليل الصف (I) على الحد الأعلى (100) المحدد فى جملة DIM .

INPUT

فى عديد من نسخ البيسك للحاسبات الدقيقة ، يمكن استخدام جملة INPUT لطباعة رسالة تلقين (أى سلسلة حروف) قبل إدخال بيانات الإدخال. ولعمل ذلك ، يجب أن تتبع الرسالة الكلمة الدالة INPUT ويجب أن تكون محصورة بين علامتى اقتباس . وغالباً مانحتاج الفاصلة المنقوطة (;) بعد الرسالة ، وذلك بغرض فصل سلسلة الحروف من قائمة متغيرات الإدخال .

مثال ٩ — ٢٠

إدرس الجملة :

10 INPUT "WHAT IS YOUR NAME"; N$

وعند تنفيذ هذه الجملة تظهر الرسالة التالية على النهاية الطرفية :

WHAT IS YOUR NAME?

ورداً على ذلك (أي سلسله الحروف التى تعطى للمتغير N$) فسوف ندخل قيمة المتغير N$ على نفس السطر وتلى مباشرة علامة الاستفهام . وبذلك ، إذا اختار المستخدم الإجابة SANTA CLAUS فتطبع هذه القيمة ويظهر السطر كاملا كالتالى :

WHAT IS YOUR NAME? SANTA CLAUS

(لاحظ أن إجابة المستخدم قد وضع تحتها خط) .

وتتضمن بعض نسخ بيسك فرصة اختيارية لعدم ظهور علامة الاستفهام بعد رسالة التلقين . ويمكن إنجاز ذلك عادة بوضع الفاصلة (،) بدلا من الفاصلة المنقوطة (;) عند نهاية رسالة التلقين .

مثال ٩ — ٢١

الآتى اختلاف فى جملة INPUT عن مثال ٩ — ٢٠ . وفى هذا المثال ستحذف علامة الاستفهام التى تتبع جملة التلقين .

10 INPUT "PLEASE ENTER YOUR NAME: ",N$

يؤدى تنفيذ هذه الجملة إلى العبارة التالية

PLEASE ENTER YOUR NAME:

للظهور على النهاية الطرفية ، فإذا استجاب المستخدم ثانية بإدخال اسم SANTA CLAUS فإن السطر كله سيظهر كالآتى :-

PLEASE ENTER YOUR NAME: SANTA CLAUS

(استجابة المستخدم تحتها خط) .

لاحظ أن عبارة التلقين قد تم تعديلها لتظهر كجملة بدلاً من سؤال حيث لا توجد علامة استفهام مرتبطة بهذا التلقين .

INKEY$ and INPUT$

تستخدم الدالتان (وليس الجملتان) INKEY$ و INPUT$ لإدخال حرف مفرد أو سلسلة حروف متعددة على الترتيب من لوحة المفاتيح ، وحتى يمكن استخدام هذه الدوال بطريقة سليمة فلابد من تخصيص سلسلة الحروف التى أدخلت إلى متغير لسلسلة الحروف المناسبة . واستخدام هذه الدوال لا يولد علامة استفهام عندما تطلب بيانات مدخلة وذلك لا يماثل جملة INPUT . وسلسلة الحروف المدخلة تدخل

ببساطة من لوحة المفاتيح بدون الضغط على مفتاح الرجوع وسوف لا تعرض الحروف المدخلة على النهاية الطرفية .

وتستخدم أحياناً الدالتان INPUT$, INKEY$ لإيقاف تنفيذ البرنامج ، وعلى ذلك فإن تنفيذ البرنامج سيتوقف مؤقتاً عند مقابلة إحدى هاتين الدالتين حتى يتم إدخال سلسلة الحروف المطلوبة.وتحتاج INKEY$ لسلسلة حروف فردية ، بينما تحتاج INPUT$ لسلسلة حروف يتحدد طولها كجزء من الدالة المشار إليها . ويحتوى قسم ٩ ــ ٣ على مناقشة أكثر عمومية على الدوال فى بيسك الحاسبات الدقيقة .

مثال ٩ ــ ٢٢

يحتوى برنامج بيسك للحاسبات الدقيقة على عدد من جمل PRINT يقصد بها إمداد مستخدمى البرنامج بمجموعة من التعليمات . وسوف تملأ هذه التعليمات شاشة TV كاملة (ومن ثم تختفى فور مسح الشاشة) . ومن أجل اعطاء المستخدم وقتاً كافياً لقراءة هذه التعليمات فإن جمل الطباعة تتبعها جملة لاستخدام دالة INKEY$ ، ومن ثم فإن التعليمات ستبقى على الشاشة حتى يضغط المستخدم على مفتاح مسبباً إدخال سلسلة حروف مفردة إلى الحاسب .

مبين فيما يلى جزء من البرنامج .

```
10 PRINT TAB (34); "INSTRUCTIONS"
    :
190 PRINT TAB(26); "(PRESS ANY KEY TO CONTINUE)"
200 A$ = INKEY$ : IF A$="" THEN 200
```

لاحظ أن البرنامج سوف يستمر فى الدوران فى الحلقة خلال جملة 200 حتى يتم الضغط على أحد المفاتيح الذى يتسبب فى إدخال سلسلة الحروف الفردية المطلوبة والتى يتم تخصيصها للسلسلة .

وهنا طريقة أخرى لإنجاز نفس الشيء باستخدام دالة INPUT$.

```
20 PRINT TAB (34); "INSTRUCTIONS"
    :
190 PRINT TAB(26); "(PRESS ANY KEY TO CONTINUE)"
200 A$=INPUT$(1)
```

فى هذه الحالة سيصل البرنامج إلى حالة توقف حتى يتم إدخال سلسلة حروف فردية من لوحة المفاتيح (1 الذى يظهر بين القوسين بعد INPUT$ يعنى أن السلسلة المدخلة تحتوى على حرف واحد) . وسنتناول باستفاضة هذه الأنواع من طرق البرمجة فى الفصلين العاشر والحادى عشر .

PRINT USING

جملة PRINT USING متاحة فى عديد من نسخ البيسك للحاسبات الدقيقة وهى تسمح بطباعة المخرجات فى صيغة معينة ، وبذلك تصف شكل ومكان كل عنصر من البيانات . ويمكن صياغة كل من البيانات العددية والحرفية ، إلا أن هذه الخاصية عامة تكون أكثر فائدة فى البيانات الرقمية .

وتوجد عدة طرق مختلفة لصياغة بيانات الإخراج بواسطة جملة PRINT USING. وتتضمن كل هذه الطرق وضع صيغة سلسلة حروف بعد الكلمة الدالة PRINT USING مباشرة وقبل قائمة بنود المخرجات . ويجب أن تظهر الفاصلة المنقوطة (;) بين صيغة سلسلة الحروف وبين أول بند من المخرجات .

مثال ٩ ـــ ٢٣

وإليك أكثر الصيغ شيوعاً لجملة PRINT USING

```
100 PRINT USING "##.## ";A,B,C
```

فى هذا المثال سلسلة الحروف المصاغة هى " ##.## " حيث تصف حقلا رقمياً يحتوى على علامة عشرية من خانتين فقط على جانبى العلامة العشرية .وسوف تقرب الكسور (الأرقام العشرية) التى تمتد إلى ما وراء نطاق الخانتين . والمكان الخالى فى نهاية سلسلة الحروف المصاغة يجب أن يكون موجوداً وذلك ليفصل بين القيم المطبوعة لكل من A و B و C .

والآن نفرض أن قيم المتغيرات A و B و C هى 17.667 و 5.38 – و 40 على الترتيب ، فسوف يتم توليد سطر المخرجات التالى باستخدام جملة PRINT USING التى ظهرت أعلاه :

```
17.67     -5.38     40.00
```

مثال ٩ ـــ ٢٤

صيغة أخرى لجملة PRINT USING مبينة فيما يلى :

```
200 PRINT USING "##.###^^^^"; VALUE
```

لاحظ الأسهم الأربعة الرأسية عند نهاية سلسلة الحروف المصاغة وهى تصف الترميز الأسى . وبذلك إذا كانت قيمة المتغير VALUE ممثلة بالرقم 856.07 فإن جملة PRINT USING السابقة سوف تولد القيمة :

```
8.561E+02
```

وبالمثل إذا كانت قيمة المتغير VALUE هى الرقم 8560.7 – فإن جملة PRINT USING سوف تولد :

```
-8.561E+02
```

مثال ٩ ـــ ٢٥

هذا المثال يوضح كيف يمكن إضافة الفصلات (,) فى أى رقم من المخرجات .

```
30 PRINT USING "########, . ##"; COST#
```

الفاصله التى تسبق العلامة العشرية فى سلسلة الحروف المصاغة سوف تسبب وضع الفاصله (,) فى جزء الرقم الصحيح لكل كمية من المخرجات، وبذلك إذا كان المتغير المزدوج الدقة #COST (قيمته هى 15673088.209) فإن الجملة السابقة سوف تولد المخرج التالى :

```
15,673,088.21
```

لاحظ أن الجزء الصحيح من الرقم قد تم تقسيمه إلى مجموعات من ثلاث خانات مبتدئاً من العلامة العشرية ومتحركاً منها إلى اليسار .

وهذه الخاصية مفيدة فى التقارير المالية وبعض أنواع التطبيقات العملية الأخرى .

توجد اختلافات عديدة فى جملة PRINT USING حيث أن لها تفاصيل كثيرة جداً عرضها فى هذا الكتاب غير عملى . وننصح القارىء أن يرجع فى ذلك إلى الكتيب الخاص ببرمجة الحاسب الذى يقتنيه أو يستخدمه .

LPRINT and LPRINT USING

تتضمن بعض نسخ بيسك الحاسبات الدقيقة على جملة LPRINT وهى تستخدم فى طباعة بيانات الإخراج على آلة الطباعة أو على نهاية طرفية طابعة (بدلا من مراقبة ذلك على شاشة العرض TV) . والجملة مطابقة تماماً لجملة PRINT ماعداً أننا نستخدم الكلمة الدالة LPRINT بدلا من PRINT .

وهذه النسخ من بيسك لحاسبات الدقيقة والتى تستخدم جملة PRINT USING يمكن أيضاً أن تتضمن جملة LPRINT USING المماثلة .

مثال ٩ ــ ٢٦

يحتوى برنامج بيسك على جملتى الطباعة التاليتين :

```
200 PRINT A,B,C
210 LPRINT A,B,C
```

وسوف تسبب الجملة الأولى عرض قيم المتغيرات A و B و C على شاشة العرض TV ، بينما الجملة الثانية تسبب طباعة نفس القيم على نهاية طرفية طابعة .

تعليقات عامة General Comments

نذكر القارىء أن الجمل التى تم عرضها فيما سبق تهدف إلى أن تكون ممثلة لنسخ بيسك الحاسبات الدقيقة المتاحة عموماً . وقد تكون متاحة أو غير متاحة فى نسخة معينة من اللغة ، وإذا كانت متاحة فقد تكون التفاصيل مختلفة نوعاً ما ، ومعظم نسخ بيسك الحاسات الدقيقة تتضمن أيضاً جملاً إضافية لم تتم مناقشتها . ويحتوى ملحق (د) على ملخص أكثر شمولاً للجمل المتاحة عموماً فى معظم نسخ ميكروسوفت بيسك . ويجب أن يرجع القارىء مرة أخرى إلى المرجع الخاص بالآلة التى يستخدمها لتفصيلات ومعلومات اكثر عن اللغة المستخدمة .

مثال ٩ ــ ٢٧ البحث عن نهاية عظمى على الحاسب الدقيق Search for a Maximum on a Microcomputer

نفرض إننا رجعنا إلى مثال البرمجة الذى اعتبرناه فى مثال ٦ــ٦،والآن سنستخدم عددًا من الخصائص الجديدة المتاحة على ميكروسوفت بيسك كما هى منفذة على حاسب IBM الشخصى . ويمثل شكل ٩ــ٣ نسخة من برنامج الحاسب الدقيق . والمنطق المستخدم هو أساساً نفس المعروض فى مثال ٦ــ٦ رغم أن أسماء المتغيرات قد تم تمديدها وتم استخدام بعض البرامج المختلفة التى تمت كتابتها سابقاً ، وبالتحديد فإن المتغيرات الذى سبق تسميتها X4,X3,X2,X1 أصبحت تسمى Right, YR,YL, left على الترتيب ، وأيضاً فإن Y2 و Y3 أصبحت تسمى YL و YR . و D أصبحت تسمى DISTANCE ، و I أصبحت تسمى COUNT% .وأسماء المتغيرات هذه تم تغيرها حتى تكون أكثر وصفاً لحقيقة المسألة . والبرنامج هو أكثر عمومية نوعاً ما عن ذلك الموضح فى مثال ٦ ــ ٦ من حيث أن أكبر عدد من العمليات المكررة هى الآن متغير مدخل (MAXCOUNT%) بدلاً من مقدار ثابت موجود فى البرنامج .

هذا بالإضافة إلى هذا البرنامج يحتوى على عدد من الجمل الجديدة التى سبق وصفها حالاً . وعلى الأخص فإننا نخص DEFSNG (سطر 30) IF—THEN الممتد (سطر 160) IF—THEN—ELSE المتداخلة (سطر 230) حلقة WHILE—WEND (سطر 150 إلى 240) ، امتداد استخدام جملة INPUT (سطر 70 إلى 100) وجملة PRINT USING (سطر 290 إلى 300) ، واستخدام هذه الجمل ينتج عنه برنامج أقصر وأكثر اختصاراً من البرنامج الموضح سابقاً فى شكل (٦ ــ ٥) .

```
10 REM    SEARCH FOR A MAXIMUM OF THE FUNCTION  Y = X * COS(X)
20 '
30 DEFSNG A-Z
40 DEF FNY(X) = X*COS(X)
50 '
60 PRINT "Search for a Maximum of the Function  y = x * cos(x)" : PRINT
70 INPUT "Left boundary : ",LEFT
80 INPUT "Right boundary: ",RIGHT
90 INPUT "Minimum distance between interior points : ",DISTANCE
100 INPUT "Maximum number of iterations : ",MAXCOUNT% : PRINT
110 COUNT% = 0
120 '
130 REM    BEGIN LOOP
140 '
150 WHILE (RIGHT - LEFT) > 3 * DISTANCE
160     IF COUNT% = MAXCOUNT% THEN PRINT "Too many iterations" : END
170     COUNT% = COUNT% + 1
180     XL = LEFT + .5 * (RIGHT - LEFT - DISTANCE)
190     XR = XL + DISTANCE
200     YL = FNY(XL)
210     YR = FNY(XR)
220     GOSUB 360
230     IF YL = YR THEN 280 ELSE IF YL < YR THEN LEFT = XL ELSE RIGHT = XR
240 WEND
250 '
260 REM   COMPUTE AND PRINT FINAL SOLUTION
270 '
280 X = .5 * (XL + XR)
290 PRINT : PRINT "Xmax = "; : PRINT USING "##.###### ";X;
300 PRINT SPC(12);"Ymax = "; : PRINT USING "##.###### ";FNY(X)
310 PRINT : PRINT "Number of iterations = ";COUNT%
320 END
330 '
340 REM   SUBROUTINE TO PRINT THE RESULTS OF EACH ITERATION
350 '
360 PRINT : PRINT ,YL,YR
370 PRINT LEFT,XL,XR,RIGHT
380 RETURN
390 END
```

شكل ٩ – ٣

وهناك بعض الخصائص الأخرى لهذا البرنامج والتى لابد من الإشارة إليها . لاحظ أن العديد من السطور خال إلّا من الفاصلة العليا . وهذه الفاصلة العليا هى طريقة أخرى لكتابة ملاحظة فى ميكروسوفت بيسك (كل ما يتبع الفاصلة العليا يعتبر ملاحظة) وفى المثال الحالى تكون هذه الملاحظات هى طريقة فقط لادخال سطور خالية حتى تفصل الأقسام المختلفة للبرنامج (لن يقبل المفسر سطوراً مرقمة خالية بالكامل) .

بالإضافة إلى ذلك يجب ملاحظة أن جملة END تظهر فى ثلاثة أماكن مختلفة داخل البرنامج (سطر 390,320,160) وفى السطرين الأولين تستخدم جملة END بدلاً من جملة STOP حتى توقف عبارة غير مرغوب فيها تولدها STOP . وفى السطر الأخير تستخدم END بالطريقة التقليدية المعتادة ، ومن الغريب أنه لا توجد حاجة إلى END فى نهاية برنامج ميكروسوفت بيسك . وعلى ذلك فإن ظهورها فى نهاية البرنامج غير ضرورى .

شكل ٩–٤ يبين المخرجات المولدة عند تنفيذ البرنامج باستخدام معالم الإدخال والتى هى أساساً نفسها المستخدمة فى مثال ٦–٦ (رغم السماح لعدد أقل من العمليات المكررة هذه المرة ، حيث أننا نعرف عددها الذى نحتاج إليه) . وفى العادة فإن هذه المخرجات ستظهر على شاشة TV ولاحظ أن البيانات المدخلة تحتها خط .

ومن المفيد مقارنة هذه المخرجات بتلك الموضحة بشكل ٦–٦ .

```
Search for a Maximum of the Function  y = x * cos(x)

Left boundary : 0
Right boundary: 3.14159
Minimum distance between interior points : 0.0001
Maximum number of iterations : 20

                 8.063497E-05                    -7.647047E-05
 0               1.570745       1.570845          3.14159

                 .5553565       .5553716
 0               .7853725       .7854725          1.570845

                 .4508655       .4507949
 .7853725        1.178059       1.178159          1.570845

                 .5454382       .5454121
 .7853725        .9817156       .9818156          1.178159

                 .5605342       .5605293
 .7853725        .8835441       .8836441          .9818156

                 .5604048       .5604101
 .7853725        .8344583       .8345583          .8836441

                 .5610945       .5610948
 .8344583        .8590012       .8591012          .8836441

                 .5609718       .5609695
 .8590012        .8712727       .8713727          .8836441

                 .5610725       .5610713
 .8590012        .865137        .865237           .8713727

                 .5610933       .5610929
 .8590012        .8620691       .8621691          .865237

                 .5610963       .5610963
 .8590012        .8605352       .8606352          .8621691

                 .561096        .5610961
 .8590012        .8597682       .8598682          .8606352

                 .5610963       .5610964
 .8597682        .8601517       .8602518          .8606352

                 .5610963       .5610963
 .8601517        .8603434       .8604435          .8606352

Xmax =  0.860393            Ymax =  0.561096

Number of iterations =  14
```

شكل ٩ – ٤

٩ــ٣ دوال مكتبية إضافية ADDITIONAL LIBRARY FUNCTIONS

تتضمن معظم نسخ بيسك الحاسبات الدقيقة عدداً من الدوال المكتبية الخاصة ، بالإضافة إلى الدوال الموجودة أصلاً فى النسخ التقليدية من اللغة . وقد سبق فعلاً أن ناقشنا حالتين جديدتين منها INKEY%—INPUT% فى القسم ٩ــ٢ . والعديد من هذه الدوال الأكثر استخداماً فى الحاسبات الدقيقة ملخصة فيما يلى.. لاحظ أن معظم هذه الدوال تمدنا بإمكانيات عديدة متعلقة بسلاسل الحروف .

الدالة	الغرض	مثال
CDBL(*e*)	تحول قيمة التعبير العددى e إلى مزدوج الدقة	Y# = CDBL(3*X!−2*Y!)
CINT(*e*)	تحول (بالتقريب) قيمة التعبير العددى e إلى عدد صحيح	Y% = CINT(3*X!−2*Y!)
CSNG(*e*)	تحول قيمة التعبير العددى e إلى عدد مفرد الدقة .	Y! = CSNG(2*A#/B#)
FRE(*n*)	تعيد عدد البايت الغير مستخدمة فى الذاكرة (n هو متغير وهمى) .	PRINT FRE(0)
INKEY$	تعيد حرف من لوحة المفاتيح (الحرف لن يتم عرضه) .	Y$ = INKEY$
INPUT$(*n*)	تعيد متغير سلسلة حروف من n حرف من لوحة المفاتيح (سلسلة الحروف لن يتم عرضها) .	Y$ = INPUT$(3) (تعيد سلسلة حروف من ٣ حروف) .
INSTR(*s1,s2*)	تعيد الموضع حيث توجد سلسلة حروف (*s2*) داخل سلسلة حروف أخرى (*s1*)	Y = INSTR(A$,B$) (إذا كان A$ = "BASIC" و B$ = "AS", Y = 2)
LEFT$(*s,n*)	تعيد n حرف فى أقصى الشمال من سلسلة الحروف S	Y$ = LEFT$(A$,4) (إذا كان A$ = "COMPUTER" فإن Y$ = "COMP")
MID$(*s,m,n*)	تعيد جزء من سلسلة الحروف S بطول n من الحروف ابتداءً من الحرف الذى رقمه M	Y$ = MID(A$,3,5) (إذا كان A$ = "COMPUTER" فإن Y$ = "MPUTE")
LEN(*s*)	تعيد عدد الحروف فى سلسلة الحروف S	Y = LEN(A$) (إذا كان A$ = "COMPUTER" فإن Y = 8)
RIGHT$(*s,n*)	تعيد n من الحروف فى أقصى اليمين من سلسلة الحروف S	Y$ = RIGHT$(A$,4) (إذا كان A$ = "COMPUTER" فإن Y$ = "UTER")
SPACE$(*n*)	تعيد سلسلة n من المسافات الفارغة	PRINT X;SPACE$(5);Y (قيمتى X and Y سوف تفصل بخمس مسافات فارغة)
STR$(*e*)	تعيد تمثيل سلسلة حروف من التعبير العددى e	Y$ = STR$(K + 1) (إذا كان K=5 فإن Y$ = "6")
STRING$(*m,n*)	تعيد متغير حرفى m حرف من سلسلة حروف لها كود ASCII هو n (n لايمكن أن تزيد عن 255)	Y$ = STRING$(8,42) (لأن Y$ = "********" CHR$(42) = "*")
VAL	تعيد التمثيل العددى من سلسلة الحروف S بفرض أن S تتكون من اعداد مسبوقة باشارة إختيارية) .	Y = VAL(N$) (إذا كان N$ = "900" فإن Y = 900)

وسوف تناقش بعض الدوال المكتبية الاضافية المستخدمة مع الحاسبات الدقيقة عند عرض موضوعات أخرى ، ويحتوى الملحق (د) على ملخص لكل الدوال المكتبية فى ميكروسوفت بيسك المستخدمة عموماً .

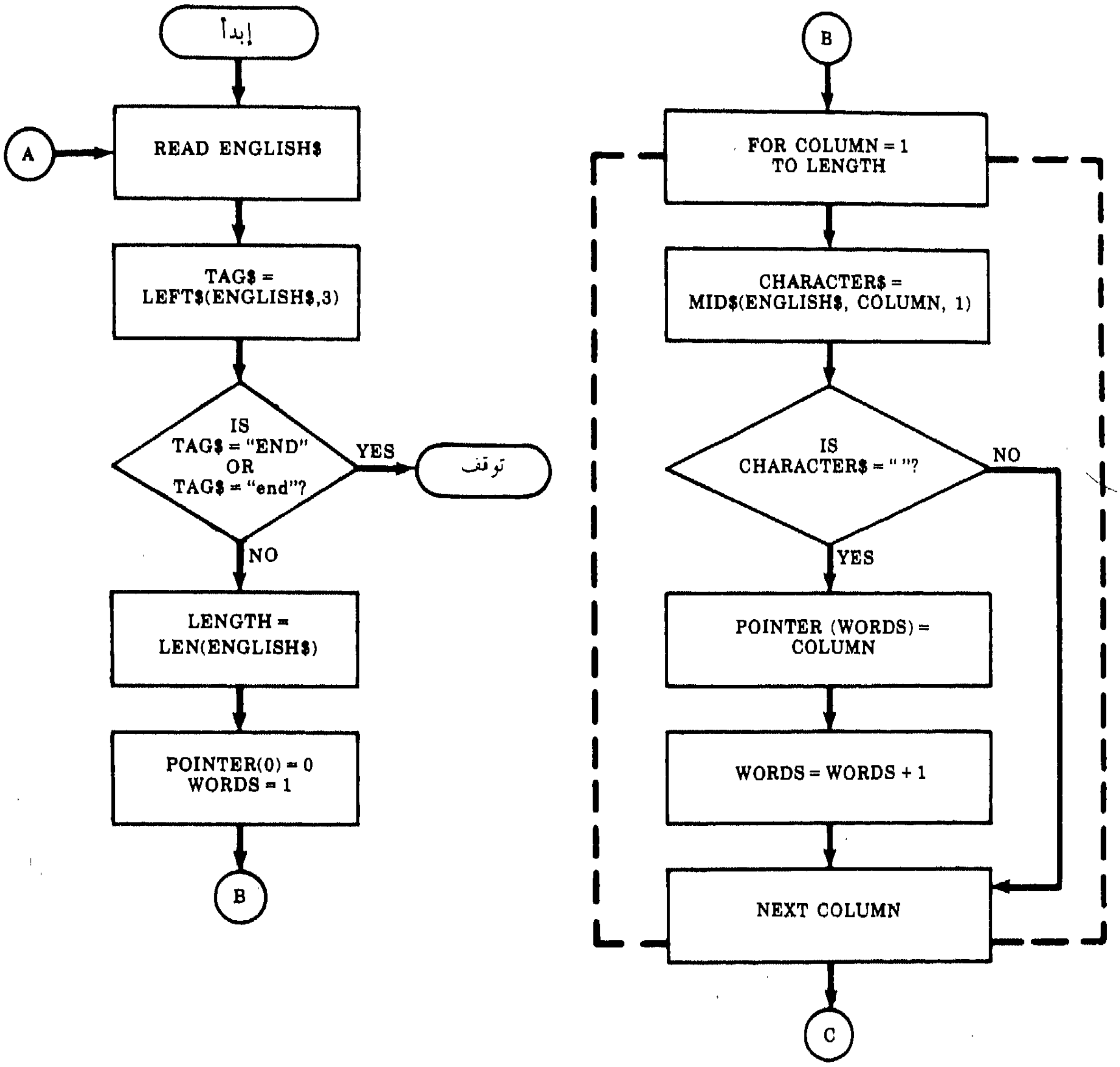
مثال ٩ ـــ ٢٨ توليد بيجلاتين على حاسب دقيق Generating Piglatin on a Microcomputer

فى مثال ٦ــ١٥ كتبنا برنامج بيسك لتوليد بيجلاتين من سطر فى نص باللغة الإنجليزية . دعنا الآن نعيد هذا المثال مستخدمين العديد من الدوال الفريدة فى ميكروسوفت بيسك .

المخطط التمهيدى للبرنامج . The Program Outline

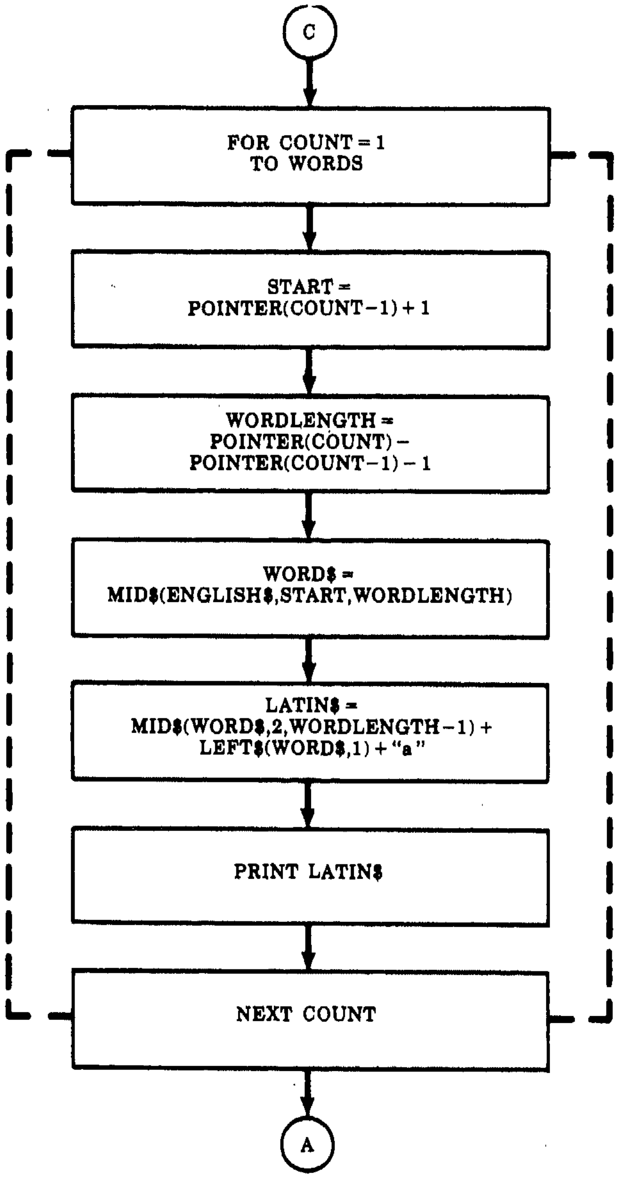
حتى يمكننا القيام بأحسن استخدام لهذه الدوال الجديدة سنستخدم استراتيجية مختلفة عن السابق استخدامها فى مثال ٦ــ١٥ . وقبل وصف الطريقة الحسابية ، سنقوم بإدخال المتغيرات :

- **ENGLISH$** = سلسلة حروف للسطر المعطى بالنص باللغة الإنجليزية .
- **TAG$** = الحروف الثلاثة الأولى فى السطر الأصلى من النص باللغة الانجليزية .
- **POINTER(I)** = صف من العناصر التى تمثل موقعاً فى الفراغ بعد الكلمة التى ترتيبها I فى السطر المعطى من النص باللغة الإنجليزية .
- **LENGTH** = عدد الحروف فى السطر المعطى من النص باللغة الإنجليزية .
- **WORDS** = عدد الكلمات فى السطر المعطى من النص باللغة الإنجليزية .
- **COLUMN** = عداد يبين موقع العمود فى السطر المعطى من النص باللغة الإنجليزية .
- **CHARACTER$** = الحرف الموجود فى عمود معين فى السطر المعطى من النص باللغة الإنجليزية .
- **COUNT** = عداد الكلمة للسطر من النص باللغة الإنجليزية .
- **WORD$** = سلسلة حروف تمثل كلمة فى السطر المعطى من النص باللغة الإنجليزية .
- **START** = رقم العمود الذى يبين البداية (الحرف الأول) للتكملة فى السطر المعطى من النص باللغة الإنجليزية .
- **WORDLENGTH** = الطول (عدد الحروف) لكلمة فى السطر المعطى من النص باللغة الإنجليزية .
- **LATIN$** = سلسلة حروف تمثل بيجلاتين المكافىء لكلمة فى السطر المعطى من النص باللغة الإنجليزية .



شكل ٩ — ٥

تكملته فى الصفحة التالية



شكل ٩ ــ ٥ (تكمله)

ويمكن وصف الحسابات الآن كما يلي :-

١ ــ اقرأ سطرًا من النص باللغة الانجليزية ، وخصص له ENGLISH $

٢ ــ اختبر شروط الوقوف لتحديد ما إذا كانت الحروف الثلاثة الأولى من النص باللغة الإنجليزية مثل :

TAG$ = LEFT$(ENGLISH$,3)

هى «END» أو «end» . وإذا كان كذلك ، فقم بإنهاء الحسابات ، وإلا فأكمل كالآتى

٣ ــ أوجد نهاية كل كلمة ، وقم بعدّ عدد الكلمات باستخدام الطريقة التالية :

(أ) حدد موضع الفراغ الخالى بعد كل كلمة مفردة على سطر النص باللغة الإنجليزية ، ثم خصص قيمته لـ POINTER ،وعلى ذلك

فإن كل قيمة لـ POINTER ستحدد نهاية الكلمة المناظرة . فمثلاً (1) POINTER ستحدد نهاية الكلمة الأولى (بداية الكلمة الثانية) ، (2) POINTER ستحدد نهاية الكلمة الثانية (بداية الكلمة الثالثة) الخ ... لاحظ أن (0) POINTER ستخصص لها قيمة 0 مشيرة إلى بداية الكلمة الأولى وذلك مباشرة بعد العمود صفر .

(ب) كل مرة تجد فيها فراغاً ؛ زد عداد الكلمة (WORDS=WORDS+1 حيث WORDS خصص لها أساساً قيمة 1) .

٤ — استخرج كل كلمة من السطر الأصلى من النص وحولها إلى بيجلاتين ويمكن أن يتم ذلك كما يلى :—

(أ) ابدأ بموضع واحد بعد القيمة الأخيرة من POINTER إلى

START = POINTER(COUNT−1) + 1)

(ب) حدد طول الكلمة كالآتى :—

WORDLENGTH = POINTER(COUNT) − POINTER(COUNT − 1) − 1

(جـ) استخرج الكلمة الإنجليزية :

WORD$ = MID$(ENGLISH$,START,WORLDLENGTH)

(د) كون كلمة بيجلاتين المكافئة كالآتى :—

LATIN$ = MID$(WORD$,2,WORDLENGTH−1) + LEFT$(WORD$,1) + "a "

٥ — اطبع كلمة البيجلاتين .

٦ — استمر حتى يتم تحويل كل الكلمات فى السطر المعطى بالنص ويتم طبعها ، ثم ارجع ثانياً للخطوة ١ وابدأ من جديد . ويبين شكل ٩—٥ خريطة سير عمليات هذه الطريقة .

برنامج البيسك **The BASIC Program**

يحوى الشكل ٩—٦ برنامج البيسك الحقيقى . لاحظ استخدام العديد من الدوال المكتبية الجديدة وهى LEFT$ (السطور 100 إلى 290) ، LEN (السطر 120) ، MID$ (السطور 290,280,180). لاحظ أيضاً استخدام عمليات التسلسل فى السطر 290 . وفى النهاية لاحظ أن ملاحظات البرنامج تستخدم كلها الفاصلة العليا بدلاً من جملة REM (السطور 230,140,60,10) .

وطول هذا البرنامج تقريباً هو نفس طول مولد بيجلاتين المكتوب بلغة البيسك التقليدية ، كما هو موضح فى شكل ٦—٩ (كجزء من مثال ٦—١٥ . وعلى ذلك فإن استخدام الخصائص الجديدة للحاسبات الدقيقة لم ينقص من طول البرنامج بشكل معنوى . والمنطق خلف البرنامج الحالى هو أنه أكثر مباشرة للمثال . ولابد للقارىء أن يقوم — بحرص — بمقارنة البرنامجين على هذا الأساس .

والشكل ٩—٧ يبين المخرج الذى يتولد عند تنفيذ البرنامج . لاحظ أن النص — باللغة الإنجليزية — الذى تم ادخاله كمدخل هو نفسه كما فى مثال ٦—١٥ باستثناء استخدام كل الحروف العليا والمنخفضة (البيانات المدخلة مرة أخرى تحتها خط) ولابد للقارىء أن يقارن شكلى ٩—٧ و٦—١٠ ليرى تناظرهما .

```
10 '*** MICROCOMPUTER PIGLATIN GENERATOR ***
20 '
30 DIM POINTER(80)
40 PRINT "Welcome to Microcomputer Piglatin" : PRINT
50 '
60 '*** READ A LINE OF ENGLISH TEXT
70 '
80 PRINT "Enter a line of text below: "
90 INPUT "",ENGLISH$
100 TAG$ = LEFT$(ENGLISH$,3)
110 IF TAG$ = "END" OR TAG$ = "end" THEN END
120 LENGTH = LEN(ENGLISH$)
130 '
140 '*** FIND THE END OF EACH WORD AND COUNT THE NUMBER OF WORDS
150 '
160 POINTER(0) = 0 : WORDS = 1
170 FOR COLUMN = 1 TO LENGTH
180     CHARACTER$ = MID$(ENGLISH$,COLUMN,1)
190     IF CHARACTER$ = " " THEN POINTER(WORDS) = COLUMN : WORDS = WORDS + 1
200 NEXT COLUMN
210 POINTER(WORDS) = LENGTH + 1
220 '
230 '*** CONVERT EACH WORD TO PIGLATIN AND PRINT
240 '
250 FOR COUNT = 1 TO WORDS
260     START = POINTER(COUNT-1) + 1
270     WORDLENGTH = POINTER(COUNT) - POINTER(COUNT-1) - 1
280     WORD$ = MID$(ENGLISH$,START,WORDLENGTH)
290     LATIN$ = MID$(WORD$,2,WORDLENGTH-1) + LEFT$(WORD$,1) + "a "
300     PRINT LATIN$;
310 NEXT COUNT
320 PRINT : PRINT
330 GOTO 80
340 END
```

شكل ٩ – ٦

```
Welcome to Microcomputer Piglatin

Enter a line of text below:
this is a piglatin generator
hista sia aa iglatinpa eneratorga

Enter a line of text below:
WHAT SORT OF GARBLED MESSAGE IS THIS ANYHOW
HATWa ORTSa FOa ARBLEDGa ESSAGEMa SIa HISTa NYHOWAa

Enter a line of text below:
Now is the time for all good men to come to the aid of their country
owNa sia heta imeta orfa llaa oodga enma ota omeca ota heta idaa foa heirta

ountryca

Enter a line of text below:
end
```

شكل ٩ – ٧

٩ ــ ٤ ملفات البيانات للحاسبات الدقيقة MICROCOMPUTER DATA FILES

تتعرض أهم التطبيقات على الحاسب الدقيق لاستخدام ملفات بيانات تحت سيطرة برنامج بيسك . وتشجع معظم نسخ بيسك الحاسبات الدقيقة كلا من ملفات البيانات المتسلسلة والعشوائية ، بالرغم من وجود اختلافات جديرة بالاهتمام فى طريقة التوصل إلى هذه الملفات من نسخة بيسك إلى أخرى . وعلاوة على ذلك ، إجراءات إدارة البيانات المستخدمة لحاسبات الدقيقة عموماً ليست هى نفسها كإجراءات إدارة البيانات المستخدمة على الحاسبات الكبيرة التى قدمت فى الفصل الثامن . (لاحظ أن استخدام ملفات بيانات الحاسبات الدقيقة يتطلب نوعاً من أجهزة التخزين المساعدة ، مثل شريط كاسيت مغناطيسى ، والقرص المرن والقرص الصلب ، أو جزء الذاكرة من أشباه الموصلات) .

ونقدم فى هذا القسم عدة أمثلة بسيطة توضح استخدام ملفات البيانات باستخدام نسخة ميكروسوفت بيسك الموجودة على حاسب IBM الشخصى ، ولكن يجب أن يفهم القارىء أن هذه الأمثلة هى فقط لتطبيقات ملف بيانات الحاسب الدقيق . ويجب أن يستشير مخطط البرامج الكتيب الخاص بحاسبه الدقيق ليقرر إجراءات إدارة الملف المناسبة .

ملفات البيانات المتسلسلة Sequential Data Files

مع معظم الحاسبات الدقيقة ، يتكون ملف البيانات المتسلسلة من مجموعة مضاعفة من بنود البيانات ، ومرتبة بالتسلسل ، بدون اعتبار لأرقام السطور (وهذا يختلف عن ملفات البيانات المتسلسلة الموصوفة فى الفصل الثامن حيث أرقام السطور جزء مكمل للبيانات) . تنشأ مثل هذه الملفات ثم تقرأ وتعدل تحت تحكم البرنامج مباشرة . والطريقة التى تنجز بها موضحة فى المثال التالى :

مثال ٩ ــ ٢٩ إنشاء ملف بيانات متسلسل Creating a Sequential Data File

يبين شكل ٩ ــ ٨ برنامج بيسك للحاسب IBM الشخصى لإنشاء ملف بيانات متسلسل يحتوى على أسماء الطلبة ودرجات اختباراتهم المبينة فى شكل ٨ ــ ١ (لاحظ الآن أن ملف البيانات المتسلسل يجب أن تخلق بواسطة برنامج بيسك بدلا من طباعتها مباشرة إلى الحاسب كما هو موصوف فى الفصل الثامن) .

السطر 30 فى هذا البرنامج يتسبب فى فتح ملف البيانات المتسلسل المسمى SCORES فى حالة الخرج ("O") وتحدد له قناة البيانات رقم 1 . والسطران 40 و 50 يسمحان لسلسلتى الحروف TITLE$ و TERM$ بالدخول من لوحة المفاتيح . ثم بعد ذلك تكتب حروف هاتان السلسلتان الحرفيتان على ملف البيانات فى السطرين 60 و 70 . وبالمثل.. فنقل البيانات يحدث فى السطور 80 و 110 و 120. ويستمر تكرار هذا الإجراء حتى تدخل كلمة END بدلاً من اسم الطالب . وبعد ذلك يقفل ملف البيانات وتنتهى الحسابات .

وعند تشغيل البرنامج فإن عنوان السنة الدراسية والفصل الدراسى (مثل COMPUTER SCIENCE 100 و FALL 1982) يتم إدخالهما فى سطور منفصلة ويتبعهما رجوع عربة الآلة . وبالمثل بالنسبة لأسماء الطلبة .. كل اسم طالب يتبعه خمس درجات للاختبارات على سطر منفصل . والدرجات الخمس كلها تدخل على نفس السطر تفصل بين كل درجة وأخرى فصلة (,) وآخر درجة يتبعها رجوع عربة الآلة .

يبين شكل ٩ ــ ٩ محتويات ملف البيانات ينتج عند تشغيل هذا البرنامج باستخدام أسماء الطلاب ودرجات الاختبارات المعطاه فى شكل ٨ ــ ١ . ملف البيانات هذا سوف يسمى SCORES . قارن شكل ٩ ــ ٩ على ملف البيانات المناظر والمبين فى أسفل شكل (٨ ــ ١) .

يجب أن يُفهم أن هذه المعلومات سوف تكتب على وسيلة تخزين مساعدة (مثل القرص المرن) بدلاً من آلة الطباعه . وبمجرد أن يتم إنشاء ملف البيانات المتسلسل فإنه يمكن وضع محتوياته كقائمة على آلة الطباعه ، وهذا يبين كيف تم توليد شكل ٩ ــ ٩ .

والإجراء العام لتعديل ملف البيانات هو نسخ محتويات ملف البيانات القديم على ملف البيانات الجديد مع دمج أى إضافات أو تغييرات فى البيانات أثناء إجراء النسخ . وبعد الانتهاء من عملية التعديل يلغى الملف القديم "KILLed" ويعطى ملف البيانات الجديد (المعدل) اسم الملف القديم .

```
10 REM ****** CREATE A SEQUENTIAL DATA FILE ******
20 REM        (STUDENT EXAMINATION SCORES)
30 OPEN "O",1,"SCORES"
40 INPUT "Course title";TITLE$
50 INPUT "Term";TERM$
60 PRINT #1,TITLE$
70 PRINT #1,TERM$
80 PRINT: INPUT "Name";N$
90 IF N$="END" OR N$="end" THEN 140
100 INPUT "Exam scores";C1,C2,C3,C4,C5
110 PRINT #1,N$
120 PRINT #1,C1;C2;C3;C4;C5
130 GOTO 80
140 CLOSE
150 END
```

شكل ٩ — ٨

```
Comp Sci 141
Fall 1985
Adams B F
 45  80  80  95  55
Brown P
 60  50  70  75  55
Davis R A
 40  30  10  45  60
Fisher E K
 0  5  5  0  10
Hamilton S P
 90  85  100  95  90
Jones J J
 95  90  80  95  85
Ludwig C W
 35  50  55  65  45
Osborne T
 75  60  75  60  70
Prince W F.
 85  75  60  85  90
Richards E N
 50  60  50  35  65
Smith M C
 70  60  75  70  55
Thomas B A
 10  25  35  20  30
Wolfe H
 25  40  65  75  85
Zorba D R
 65  80  70  100  60
```

شكل ٩ — ٩

```
10 REM ****** PROGRAM TO PROCESS STUDENT EXAMINATION SCORES ******
20 REM ******          USING SEQUENTIAL DATA FILES           ******
30 DIM C(15)
40 OPEN "I",1,"SCORES"
50 OPEN "O",2,"UPDATE"
60 INPUT #1,TITLE$
70 INPUT #1,TERM$
80 PRINT "Course title: ";TITLE$,"Term: ";TERM$
90 PRINT #2,TITLE$: PRINT #2,TERM$
100 PRINT: INPUT "Exam number";K
110 PRINT: INPUT "Calculate averages (Y/N) ";ANS$
120 PRINT: INPUT #1,N$
130 FOR I=1 TO K-1: INPUT #1,C(I): NEXT I
140 PRINT N$,
150 INPUT "New score";C(K)
160 PRINT #2,N$
170 SUM=0
180 FOR I=1 TO K: PRINT #2,C(I);: SUM=SUM+C(I): NEXT I
190 IF ANS$="N" OR ANS$="n" THEN 230
200 AVG=SUM/K
210 PRINT "Average=";AVG
220 PRINT #2,AVG
230 IF NOT EOF(1) THEN 120
240 CLOSE
250 KILL "SCORES"
260 NAME "UPDATE" AS "SCORES"
270 END
```

شكل ٩ ــ ١٠

```
Course title: Comp Sci 141  Term: Fall 1985
Exam number? 6
Calculate averages (Y/N) ? y
Adams B F      New score? 75
Average= 71.66666
Brown P        New score? 80
Average= 65
Davis R A      New score? 55
Average= 40
Fisher E K     New score? 5
Average= 4.166667
Hamilton S P   New score? 90
Average= 91.66666
Jones J J      New score? 80
Average= 87.5
Ludwig C W     New score? 70
Average= 53.33333
Osborne T      New score? 80
Average= 70
Prince W F     New score? 100
Average= 82.5
Richards E N   New score? 70
Average= 55
Smith M C      New score? 75
Average= 67.5
Thomas B A     New score? 10
Average= 21.66667
Wolfe H        New score? 95
Average= 64.16666
Zorba D R      New score? 95
Average= 78.33334
```

شكل ٩ ــ ١١

مثال ٩ ــ ٣٠ تشغيل درجات اختبارات طالب على حاسب دقيق

Processing Student Examination Scores on a Microcomputer

نرى فى شكل ٩ ــ ١ برنامج بيسك خاص بالحاسب الدقيق راديو شاك تى . آر . إس 80 يسمح لنا بتعديل ملف البيانات المتسلسل الذى تم إنشاؤه فى المثال ٩ ــ ٢٩ ، (وهذا البرنامج يوازى برنامج بيسك لحاسب الكبير المبين فى شكل ٨ ــ ٥ ، والذى يحقق نفس الغرض).

وفى هذا البرنامج يتسبب السطر 40 فى فتح ملف البيانات القديم المسمى SCORES فى حالة إدخال النمط ("I") وإعطائه قناة البيانات (المخزن الوسيط) رقم 1 . وبالمثل.. فالسطر 50 يتسبب فى فتح الملف الجديد المسمى UPDATE فى حالة إخراج النمط ("O") وإعطائه قناة البيانات رقم 2 . ثم بعد ذلك يقرأ عنوان المادة الدراسية (TITLE$) والفصل الدراسى (TERM$) من الملف SCORES (السطران 60 و70) ثم تعرض على شاشة مراقب TV (السطر 80) وتكتب أيضاً إلى الملف UPDATE (السطر 90) يسمح السطران 100 و 110 للمستفيد أن يصف رقم الاختبار المراد حساب متوسط الدرجات لكل طالب أم لا . ثم بعد ذلك نقابل حلقة تكرارية تتراوح ما بين السطور 120 إلى 240 حيث يتم إجراء الخطوات التالية لكل طالب :

١ ــ تقرأ اسم الطالب ودرجات الاختبار السابق SCORES (السطران 120 و 130) .

٢ ــ يعرض اسم الطالب على شاشة المراقبة .TV (السطر 140) ثم يتم إدخال درجة جديدة من خلال لوحة المفاتيح (السطر150)

٣ ــ يكتب اسم الطالب ودرجات اختبار الطالب على UPDATE ثم يحسب متوسط الدرجات (اختيارياً) (السطور 160 إلى 200)

٤ ــ يعرض المتوسط المحسوب على شاشة المراقبة TV (السطر 210) ثم يكتب إلى UPDATE (السطر 220) .

٥ ــ ويستمر فى هذه العملية حتى تكتشف نهاية الملف SCORES (السطر 230) . وفى هذا الوقت نقفل الملفين (السطر 240) ، ويلغى الملف الأصلى (KILLED فى السطر 250 . وسنسمى الملف الجديد SCORES (السطر 260) .

ملفات البيانات العشوائية Random Data Files

من وجهة نظر البرمجة .. فإن ملفات البيانات العشوائية أصعب فى استخدامها شيئاً ما عن الملفات المتسلسلة ، رغم أنه يمكن التوصل إليها أسرع عند تنفيذ البرنامج.ومن ناحية أخرى فإن مركبات الملفات الفردية يمكن تداولها أسرع كثيراً فى ملف التداول العشوائى كما هو موضح فى الفصل الثامن .

وإجراءات إنشاء وقراءة وتعديل ملفات البيانات العشوائية تختلف بصورة عامة فى الحاسب الدقيق عنها فى الحاسب الكبير . فمثلاً على الحاسب IBM الشخصى يعرف ملف البيانات العشوائى كمجموعة من سجلات ذات طول ثابت . تخزن كل بنود البيانات الفردية فى السجل كسلسلة حرفية ويجب تحويل بنود البيانات التى تمثل قيما عددية إلى أرقام حقيقية عند نقلها لذاكرة الحاسب . وبالمثل ، يجب تحويل هذه البنود مرة أخرى إلى سلاسل حرفية قبل نقلها إلى ملف البيانات . ينشأ مخزن وسيط يحتوى على سجل واحد فى ذاكرة الحاسب . ويقسم المخزن الوسيط إلى حقول حيث يحوى كل حقل بندا واحداً من بنود البيانات .

مثال ٩ ــ ٣١ التحكم فى المخزون لحاسبات الدقيقة : Microcomputer Inventorv Control

يحتوى شكل ٩ ــ ١٣ برنامجاً بسيطاً للتحكم فى المخزون وهو مكتوب للحاسب IBM الشخصى.والطريقة العامة لهذه المسألة هى أساساً نفس المخطط التمهيدى للمثال ٨ ــ ١٠ ، يتشابه البرنامج لدرجة كبيرة مع النسخة المقدمة للحاسب الكبير فى شكل ٨ ــ ١٢ ومع كل فقد تم اعادة تسمية المتغيرات لتكون أكثر وصفاً للمسألة الحقيقية،وعلى الأخص رقم السجل ، يشار إليه الآن RECNO% وتسمى السلسلة الحرفية التى تمثل مستوى المخزون (أى عدد الوحدات) UNITS$ وعددها الصحيح المناظر يسمى UNITS%.ويسمى التغير فى مستوى المخزون الآن CHANGE% .

```
Comp Sci 141
Fall 1985
Adams B F
 45  80  80  95  55  75  71.66666
Brown P
 60  50  70  75  55  80  65
Davis R A
 40  30  10  45  60  55  40
Fisher E K
 0  5  5  0  10  5  4.166667
Hamilton S P
 90  85  100  95  90  90  91.66666
Jones J J
 95  90  80  95  85  80  87.5
Ludwig C W
 35  50  55  65  45  70  53.33333
Osborne T
 75  60  75  60  70  80  70
Prince W F
 85  75  60  85  90  100  82.5
Richards E N
 50  60  50  35  65  70  55
Smith M C
 70  60  75  70  55  75  67.5
Thomas B A
 10  25  35  20  30  10  21.66667
Wolfe H
 25  40  65  75  85  95  64.16666
Zorba D R
 65  80  70  100  60  95  78.33334
```

شکل ۹ — ۱۲

```
10 REM *** Microcomputer Inventory Control Program ***
20
30 OPEN "R",#1,"INVTRY",5
40 FIELD #1,5 AS UNITS$
50 PRINT "Stock numbers run from 1 to 2000"
60 PRINT: PRINT "To end session, enter a negative stock number"
70 PRINT: INPUT "Stock number";RECNO%
80 IF RECNO% < 1 THEN 260
90 IF RECNO% > 2000 THEN PRINT: GOTO 50
100
110 REM *** UPDATE A RECORD
120
130 GET 1,RECNO%
140 UNITS%=CVI(UNITS$)
150 PRINT "Original inventory=";UNITS%;" items"
160 INPUT "Change in inventory level";CHANGE%
170 UNITS%=UNITS%+CHANGE%
180 IF UNITS% < 0 THEN UNITS% = 0
190 PRINT "New inventory=";UNITS%;" items"
200 LSET UNITS$=MKI$(UNITS%)
210 PUT 1,RECNO%
220 GOTO 70
230
240 REM *** END RECORD UPDATE
250
260 CLOSE
270 END
```

شکل ۹ — ۱۳

```
Stock numbers run from 1 to 2000

To end session, enter a negative stock number

Stock number? 1186
Original inventory= 346  items
Change in inventory level? -45
New inventory= 301  items

Stock number? 708
Original inventory= 368  items
Change in inventory level? 200
New inventory= 568  items

Stock number? 84
Original inventory= 147  items
Change in inventory level? 16
New inventory= 163  items

Stock number? 1400
Original inventory= 78  items
Change in inventory level? -50
New inventory= 28  items

Stock number? -999
```

شكل ٩ ــ ١٤

٩ ــ ٥ إجراءات لغة الآلة فى البيسك MACHINE-LANGUAGE PROCEDURES IN BASIC

تتضمن بعض نسخ بيسك الحاسبات الدقيقة إمكانيات للقيام بعمل بعض الإجراءات التى تلازم عادة لغة الآلة . وتتضمن هذه الإجراءات :

١ ــ اختيار جزء من ذاكرة الحاسب .
٢ ــ فحص محتويات ذاكرة الحاسب .
٣ ــ تغيير محتويات ذاكرة الحاسب .
٤ ــ التوصل إلى برامج فرعية بلغة الآلة .

المناقشة التفصيلية لاستخدام هذه الخصائص خارج نطاق وحدود هذا الكتاب ، إلا أن هذه الإجراءات تستخدم أساساً فى تطبيقات مثل بيانيات متقدمة ، وتوليد الصوت والموسيقى ، ومداولة الخطأ ، والتحكم فى أجهزة الوحدات الخارجية .

والخواص الأكثر شيوعاً ملخصة فيما يلى .

الجملة أو الدالة	الغــــرض	مثـــال
DEF SEG=M	يختار جزء فى الذاكرة ويبدأ فى المكان M	10 DEF SEG=32768 (يختار جزء فى الذاكرة ويبدأ فى المكان 32768)
PEEK (M)	تعاد محتويات المكان M فى الذاكرة (عشرى) .	10 LET Z=PEEK (768) (تحدد قيمة Z بمحتويات المكان رقم 768)
POKE M, X	توضع قيمة X فى مكان الذاكرة M (عشرى)	20 POKE 200, 333 (توضع القيمة 333 فى مكان الذاكرة رقم 200)
CALL V	يتوصل للبرنامج الفرعى بلغة الآلة والذى يبدأ فى المكان رقم V من الذاكرة (عشرى)	20 START=1024 30 CALL START يتوصل للبرنامج الفرعى الذى يبدأ فى المكان رقم 1024 من الذاكرة)
USR (X)	تُنْقَل قيمة X إلى البرنامج الفرعى بلغة الآلة .	40 PRINT USR (12) (تطبع قيمة مولدة بواسطة برنامج فرعى بلغة الآلة،والذى يقبل الرقم 12 كعامل إدخال) .

لاحظ : أن بعض نسخ البيسك تسمح بتوصيف بداية برنامج فرعى بلغة الآلة بواسطة جملة DEF ، فمثلا 50 DEF USR = 1280 ونذكر القارىء أن CALL و *DEF SEG,* POKE عبارة عن جمل بينما PEEK و USR عبارة عن **دوال مكتبية .**

مثال ٩ — ٣٢

ادرس جملة البيسك المبينة فيما يلى :

```
200 IF PEEK(768) > 127 THEN POKE 768,127
210 CALL START
```

تتسبب الجملة الأولى (السطر 200) فى فحص محتويات المكان رقم 768 من الذاكرة (عشرى) واستبداله بالقيمة 127 إذا تعدت قيمته الأصلية الرقم 127 . وتتصل الجملة الثانية (السطر 210) بالبرنامج الفرعى بلغة الآلة الذى يبدأ فى المكان START . (لاحظ أن التنفيذ الناجح لهذا البرنامج الفرعى يتطلب أن تكون القيمة المخزنة فى المكان 768 لاتتعدى الرقم 127) .

٩ — ٦ خواص مستبعدة عن بيسك الحاسبات الدقيقة .

FEATURES EXCLUDED FROM MICROCOMPUTER BASIC

نذكر القارىء مرة أخرى بالتعبيرات فى نسخ بيسك الحاسبات الدقيقة المختلفة المتوفرة حالياً . وبالرغم من أن معظم النسخ معقدة تماماً إلا أنه يوجد قليل من الخصائص التقليدية للبيسك التى لا توجد عادة فى بيسك الحاسبات الدقيقة .وهذه تتضمن :

١ — جمل (المصفوفات) MAT (انظر الفصل السابع)

٢ — الدوال العديدة السطور (انظر القسم ٦ — ٣) .

٣ — أوامر معينة للنظام (مثال BYE و OLD و REPLACE و SCRATCH) .

٤ — تضمين أسطر خالية (رغم أن بعض نسخ اللغة تسمح بسطر يحتوى على رقم الجملة متبوعة بواسطة فاصل ، مثال ، : 100) .

ويجدر بنا أن نذكر القارىء أن البيسك ينمو ويتغير بسرعة وتزداد شعبيته كما أن الحاسبات الدقيقة تتزايد وتتكاثر بسرعة فى المنازل ، المدارس والمكتبات فى مجتمعاتنا . ولذا نتوقع خصائص جديدة متطورة فى النسخ المستقبلية من اللغة .

اسئلة للمراجعة

Review Questions

٩ — ١ ماهى الميزة الموجودة من استخدام أسماء المتغيرات الطويلة ؟

٩ — ٢ هل يمكن استخدام كلمات بيسك الدالة (مثل PRINT و DATA و NEXT) كأسماء متغيرات ؟

٩ — ٣ اذكر أنواع البيانات المختلفة المتاحة فى نسخ البيسك للحاسبات الدقيقة . كيف نميز بين أنواع المتغيرات المناظرة عن بعضها ؟

٩ — ٤ ماهو نوع النتيجة التى نحصل عليها من عملية رياضية تتضمن :

(أ) بيانات صحيحة وحقيقية ؟

(ب) بيانات صحيحة ومزدوجة الدقة ؟

(ﺟ) بيانات حقيقية ومزدوجة الدقة ؟

٩ — ٥ ماذا يحدث عندما تعطى بيانات رقمية من نوع ما إلى متغير رقمى من نوع مختلف ؟

٩ — ٦ ماذا يحدث إذا أعطينا كمية حقيقية أو مزدوجة الدقة إلى متغير صحيح ؟

٩ — ٧ كيف تختلف القسمة الصحيحة عن القسمة العادية ؟

٩ — ٨ ماهو الغرض من المعامل MOD ؟ ومع أى نوع من البيانات يجب أن يستخدم ؟

٩ — ٩ ماهو الغرض من المعاملات AND و OR و NOT ؟ وفى أى نوع من الجمل تستخدم ؟

٩ — ١٠ لخص التدرج الهرمى للمعاملات الحسابية والمنطقية والمترابطة . هل كل نسخ البيسك لحاسبات الدقيقة تستخدم هذا التدرج الهرمى بالذات ؟

٩ — ١١ ماذا يقصد بالوصل ؟ وإلى أى نوع من البيانات نطبق الوصل ؟ وما هو المعامل الذى نستخدمه دائماً لنشير إلى الوصل ؟

٩ — ١٢ ماهى المزايا فى كتابة جملتين أو أكثر على نفس السطر ؟ وكيف نميز هذه الجمل عن بعضها ؟

٩ — ١٣ ماهو الغرض من جمل DEFINT و DEFSNG و DEFDBL و DEFSTR ؟ وكيف تقارن استخدام هذه الجمل باستخدام الملاحق فى أسماء المتغيرات

٩ — ١٤ صف الاستخدام الموسع لجملة IF-THEN فى معظم نسخ البيسك المحسنة .

٩ — ١٥ صف جملة IF-THEN-ELSE وقارن استخدامها باستخدام جملة IF-THEN الموسعة . وما هو الشىء المعنوى الذى أضافته عبارة ELSE ؟

٩ – ١٦ ماهو المقصود بالبرمجة الهيكلية ؟

٩ – ١٧ ماهو الغرض من جملة ON-GOSUB ؟ وكيف تختلف عن جملة ON-GO TO ؟

٩ – ١٨ ماهو الغرض من جملة ON ERROR GO TO ؟ وماهى الجملة الأخرى المستخدمة مقترنة بجملة ON ERROR GO TO ؟

٩ – ١٩ صف جملتى WHILE و WEND . وما هو النوع الجديد من البرنامج الذى يمكن أن يعرف بواسطة هاتين الجملتين ؟

٩ – ٢٠ إشرح الاستخدام لجملة INPUT فى معظم نسخ البيسك المحسنة .

٩ – ٢١ اشرح الاستخدام لكل من الدوال INPUT$, INKEY$ لخلق وقفات فى تنفيذ البرنامج . وهل كل نسخ بيسك الحاسبات الدقيقة تستخدم هاتين الدالتين ؟

٩ – ٢٢ لخص أكثر الخصائص المشتركة الموجودة فى جملة PRINT USING . وكيف تنفذ هذه الخصائص ؟ وإلى أى حد يكون لهذه الخصائص معنى ؟

٩ – ٢٣ ماهو الغرض من جمل LPRINT و LPRINT USING ؟

٩ – ٢٤ صف الغرض من كل من الدوال المكتبية التالية : CDBL, CINT, CSNG, FRE, INKEY$, INPUT$, INSTR, LEFT$, LEN, MID$, RIGHT$, STR$, STRING$, VAL . كم عدد الخلاصات المطلوبة لكل دالة ؟ وما هو نوع الخلاصات المطلوبة ؟

٩ – ٢٥ ماهو الفرق بين ملفات البيانات عشوائية والمتسلسلة كما تنفذ على الحاسب الدقيق ؟ وماهى مزايا وعيوب كل منها ؟

٩ – ٢٦ كيف يختلف استخدام ملفات البيانات المتسلسلة على الحاسب الدقيق عن استخدام ملفات البيانات المتسلسلة على حاسب كبير (كما هو موصوف فى الفصل الثامن) ؟

٩ – ٢٧ كيف يختلف استخدام ملفات البيانات العشوائية على الحاسب الدقيق عن استخدام ملفات البيانات العشوائية على حاسب كبير (كما هو موصوف فى الفصل الثامن) ؟

٩ – ٢٨ صف الغرض لكل من الجمل أو الدوال التالية : DEF SEG, PEEK, POKE, CALL, USR. ماهى الجمل وماهى الدوال ؟ وفى أى نوع من التطبيقات تستخدم هذه الجمل والدوال .

٩ – ٢٩ اذكر أسماء خواص البيسك « القياسى » أو المعيارى (المنفذ على الحاسبات الكبيرة) وغير المتاحة فى معظم نسخ بيسك الحاسبات الدقيقة .

مسائل تكميلية

Supplementary Problems

« المسائل » التالية متعلقة بجمع المعلومات أكبر منها لحل مسائل حقيقية . أجب عن الأسئلة التى تخص نسخة بيسك الحاسبات الدقيقة التى تتعلق بالحاسب الخاص بك .

٩ – ٣٠ هل تقبل نسخة البيسك الخاصة بك استخدام أسماء المتغيرات الطويلة ؟ وإن كان ذلك صحيحاً ، ماهو أقصى طول مسموح به لكل من أسماء المتغيرات ؟

٩ – ٣١ ماهى أنواع البيانات الرقمية المسموح بها ؟ وماهو الاختلاف الموجود بين الأنواع المختلفة من المتغيرات الرقمية ؟ هل يمكن لكل المتغيرات المشتركة فى الحرف الأول أن تعرف وتكون من نوع معين من البيانات ؟

٩ – ٣٢ هل يمكن القيام بعمليات حسابية بين أنواع مختلفة من بيانات رقمية ؟ وما هو نوع النتيجة التى نحصل عليها مع كل تركيبه للبيانات ؟

٩ – ٣٣ هل يمكن إعطاء نوع بيانات معين لمتغير رقمى من نوع آخر ؟ وماهى نوع النتيجة التى نحصل عليها ؟

٩ – ٣٤ ماهى المعاملات الإضافية المتاحة ؟ وما هو الغرض من كل ؟

٩ – ٣٥ ماهو التدرج الهرمى الكامل للمعاملات الحسابية والمترابطة والمنطقية ؟

٩ – ٣٦ هل تقبل نسخة البيسك الخاصة بك الوصل ؟ وإذا كان ذلك صحيحاً ، ماهو المعامل المستخدم لهذا الغرض ؟

٩ – ٣٧ هل يمكن وضع عدة جمل على سطر واحد ؟ وكيف يمكن فصل هذه الجمل عن بعضها ؟

٩ – ٣٨ كيف تنفذ جملة IF–THEN على نسخة البيسك الخاصة بك ؟ هل يمكن أن تتبع THEN عدة جمل ؟ وهل عبارة ELSE متاحة ؟

٩ – ٣٩ هل جملة ON–GOSUB متاحة على نسخة البيسك الخاصة بك ؟

٩ – ٤٠ هل جملة ON ERROR GOTO متاحه على نسخة البيسك الخاصة بك ؟ وهل جملة RESUME متاحه ؟

٩ – ٤١ هل تسمح نسخة البيسك الخاصة بك خاصية التكرار المشروط ؟ وكيف يعرف التكرار المشروط ؟

٩ – ٤٢ هل يمكن توليد رسالة تلقين بواسطة جملة INPUT فى نسخة البيسك الخاصة بك ؟ هل تلغى (لاتظهر) علامة الاستفهام ؟

٩ – ٤٣ هل الدوال INPUT$, INKEY$ متاحه على نسخة البيسك الخاصة بك ؟

٩ – ٤٤ هل تسمح نسخة البيسك الخاصة بك بجملة PRINT USING ؟ وإذا كان ذلك صحيحاً ، لخص الأنواع المختلفة لصيغ المخرجات المتاحة .

٩ – ٤٥ هل توجد جمل LPRINT, LPRINT USING على نسخة البيسك الخاصة بك ؟

٩ – ٤٦ ماهى الدوال المكتبية الإضافية المتاحة فى نسخة البيسك الخاصة بك ؟ وما هى الإمكانيات الإضافية المستمدة بواسطة هذه الدوال المكتبية ؟

٩ – ٤٧ كيف يمكن الانتفاع من ملفات البيانات المتسلسلة فى نسخة البيسك الخاصة بك ؟ كيف يمكن إنشاء ملف بيانات متسلسل ؟ وكيف يمكن إلغاؤه ؟ وكيف يمكن فتح وقفل ملفات البيانات المتسلسلة ؟ وكيف يمكن قراءة وكتابة بنود البيانات من وإلى هذه الملفات ؟

٩ – ٤٨ ماهو الإجراء العام لتعديل ملف بيانات متسلسل ؟ وما هى أوامر البيسك الخاصة والمتوافرة للقيام بإجراءات متعدده لملف بيانات متسلسلة

٩ – ٤٩ كيف يمكن الانتفاع من ملفات البيانات العشوائية فى نسخة البيسك الخاصة بك ؟ كيف يمكن إنشاء وإلغاء هذه الملفات ؟ وما هو الإجراء لتعريف كل سجل ؟

٩ – ٥٠ كيف يمكن فتح وغلق ملفات بيانات عشوائية ؟ وكيف يمكن كتابة وقراءة بنود البيانات إلى ومن هذه الملفات ؟ هل يجب تخزين البيانات الرقمية كحروف ؟ إن كان ذلك صحيحاً ، فكيف يمكن إنجاز عملية التحويل ؟

٩ – ٥١ هل يتطلب مخزن الذاكرة الوسيط عند استخدام ملفات بيانات عشوائية ؟ وإن كان ذلك صحيحاً ، هل يقسم المخزن الوسيط إلى حقول ؟ وكيف يمكن إنجاز ذلك ؟

٩ – ٥٢ هل تتضمن نسخة البيسك الخاصة بك إجراءات لغة الآلة التالية :

(أ) اختيار جزء من ذاكرة الحاسب .
(ب) فحص محتويات ذاكرة الحاسب .
(جـ) تغيير محتويات ذاكرة الحاسب .
(د) التوصل إلى برنامج فرعى بلغة الآلة .

وإن كان ذلك صحيحاً ، فكيف يمكن القيام بهذه الإجراءات ؟

٩ – ٥٣ هل توجد خواص تقليدية معينة من البيسك ومستبعدة من نسخة البيسك الخاصة بك ؟ (أنظر الملاحق A و C من أجل ملخص الخواص التقليدية للبيسك) .

٩ – ٥٤ ماهى الخواص الإضافية المعدلة المتاحة فى نسخة البيسك الخاصة بك ؟

مسائل للبرمجة

Programming Problems

٩ – ٥٥ أعد كتابة كل من البرامج التالية والتى تنتفع تماماً بالخواص المعدلة والمتاحة فى نسخة البيسك للحاسب الدقيق الخاصة بك : وعلى الأخص حاول استخدام كل من IF-THEN, IF-THEN-ELSE, WHILE/WEND , PRINT USING كلما أمكن .

(أ) جذور المعادلة التربيعية (مثال ٢ – ٣٠) .
(ب) جذور المعادلة الجبرية (مثال ٤ – ٥) .
(جـ) حساب قيمة الاستهلاك (مثال ٤ – ٩) .
(د) إيجاد متوسط بيانات تلوث الهواء (مثال ٤ – ١٦) .
(هـ) إعادة ترتيب قائمة من الأرقام (مثال ٥ – ١٤) .
(و) مولد بجلاتين (مثال ٦ – ١٥) .
(ز) لعبة صدفة أو خط (اصطياد كرابس) (مثال ٦ – ٢٠) .
(ح) محاكاة ارتداد كرة (مثال ٦ – ٢٨) .

٩ – ٥٦ وسع برنامج مرتب الكلمات المبعثرة (مثال ٥ – ٩) بحيث يتمكن من إعادة تنظيم الحروف فى أى كلمة (أى عدد من الحروف) إلى كل التوافقيات الممكنة . واستخدم كل الدوال المكتبية الموسعة المتاحة فى نسخة البيسك الخاصة بك .

٩ – ٥٧ أعد كتابة البرامج المعطاة فى الأمثلة ٩ – ٢٩ ، ٩ – ٣٠ وذلك لإنشاء ومعالجة مجموع النقاط التى يحرزها الطالب فى الامتحانات ، وذلك باستخدام إجراءات ملف بيانات متسلسل متاح فى النسخة الخاصة بك من لغة البيسك (أنظر أيضاً الأمثلة ٨ – ٢ و ٨ – ٣ و ٨ – ٤)

٩ – ٥٨ أعد كتابة برنامج تحكم المخزون المعطى فى مثال ٩ – ٣١ باستخدام إجراءات ملف بيانات عشوائى متاح فى النسخة الخاصة بك من لغة البيسك : (أنظر مثال ٨ – ١٠) .

٩ – ٥٩ أعد كتابة برنامج البحث الثنائى المعطى فى مثال ٨ – ١٣ مستخدماً إجراءات ملف بيانات عشوائى متاح فى النسخة الخاصة بلك من لغة البيسك .

٩ – ٦٠ اكتب برنامج بيسك كاملا سوف ينشأ وينتفع من ملف بيانات متسلسل يحتوى على أسماء وعناوين وأرقام تليفونات كما تم توصيفها فى المسألة ٨ – ٤٠ . استخدم إجراءات ملف بيانات متسلسل متاح فى النسخة الخاصة بلك من لغة البيسك .

٩ – ٦١ كرر المسألة ٩ – ٦٤ ولكن بالانتفاع بملف بيانات عشوائى استخدم إجراءات ملف البيانات العشوائى المتاح فى النسخة الخاصة بلك من لغة البيسك . قارن ذلك مع نسخة ملف البيانات المتسلسل وذلك من وجهة نظر سهولة البرمجة وسرعة ، التنفيذ .

٩ – ٦٢ حل المسألة ٥ – ٥٧ (د) (حساب الانحراف المعيارى لقائمة من الأرقام باستخدام معادلتين مختلفتين) باستخدام أرقام حقيقية (دقة مفردة ثم كرر الحسابات باستخدام دقة مزدوجة . قارن النتائج فى كل من الحالتين .

٩ – ٦٣ حل مسألة ٥ – ٥٥ (حساب نتائج امتحان الطلبة ، متضمناً متوسط الفصل وحياد متوسط كل طالب عن متوسط الفصل) وذلك باستخدام الخصائص المحسنة المتاحة فى النسخة الخاصة بك فى لغة البيسك للحاسب الدقيق . وعلى الأخص ، تأكد من استخدام جملة PRINT USING .

الفصل ١٠

بيئة الحاسب الدقيق .

The Microcomputer Environment

سنناقش فى هذا الفصل بيئة الحاسب الدقيق من حيث المكونات المادية (Hardware) والغير مادية (Software) ، أى البرامج الجاهزة . سنبدأ بمناقشة طرق تمييز الملفات ونظم أوامر البيسك للحاسب الدقيق ، ثم سنعتبر الخواص الوحيدة لشاشة TV ولوحة المفاتيح للحاسب الدقيق مع عدة خصائص بيسك الحاسب الدقيق ، ثم سنعتبر فيما بعد استخدام بعض أجهزة الإدخال القابلة للبرمجة واستخدام الصوت واللون ، ونختتم هذا الفصل بمناقشة مختصرة عن طرق البرامج الداخلية للتعديل .

هذا بالإضافة إلى أن اهتمامنا سيكون على استخدام ميكروسوفت بيسك ونظام التشغيل المصاحب له (MS—DOS) ، كما يتم تنفيذه على حاسب IBM الشخصى وغيره من الحاسبات الدقيقة من هذا النوع . (وتسمى شركة IBM نظام التشغيل هذا ، PC—DOS) . وعلى القارىء أن يطلع على الكتيب المرجعى المناسب للحصول على معلومات عن الأنواع الأخرى من الحاسبات الدقيقة أو نظم تشغيلها .

١٠ — ١ تمييز الملفات FILE DESIGNATIONS

يكون لأغلب الحاسبات الدقيقة جهاز تخزين ضخم واحد على الأقل ، وعادة جهازان أوثلاثة ،والتى يمكن تخزين أنواع مختلفة من الملفات عليها . والقرص المرن هو أكثر الأنواع شيوعاً لجهاز التخزين الضخم ، إلا أنه من المتاح أيضاً أشرطة كاسيت وأقراص صلبة وأقراص ضوئية وذاكرات فقاعية . ويمكن استخدام أى من أجهزة التخزين الكبيرة هذه لتخزين برامج البيسك ، وملفات البيانات والأنواع الأخرى من الملفات (مثل برامج لغة الآلة التى هى جزء من نظام التشغيل) . وعلى ذلك فعندما نشير إلى ملف معين تكون هناك حاجة لتحديد اسم الملف ونوع الملف وجهاز التخزين الضخم الذى تم التخزين عليه .

ويحتوى الملف الكامل المميز الموجود على حاسب IBM الشخصى (أو أى حاسب دقيق آخر يستخدم نظام تشغيل MS—DOS) على البنود الثلاثة التالية :—

١ — مرجع جهاز التخزين الضخم (حرف مفرد تعقبه نقطتان) .
٢ — اسم ملف (١ — ٨ حروف) .
٣ — امتداد ملف يحدد نوع الملف (١ — ٣ حروف مسبوقة بنقطة) .

مثال ١٠ — ١

الآتى يبين ملف MS—DOS كاملاً مميزاً :—

A:SAMPLE.BAS

هذا الملف المميز يشير إلى برنامج يسمى SAMPLE تم تخزينه على محرك القرص A . (ومحركات الأقراص سترقم بحروف A ، B و C) . وعلى ذلك فإن جهاز التمييز هو A ، واسم الملف هو SAMPLE والامتداد هو BAS . لاحظ النقطتين بين جهاز التمييز واسم الملف والنقطة التى تفصل اسم الملف والامتداد .

وعندما يعمل الحاسب الدقيق فإن أحد أجهزة التخزين الضخمة تكون دائماً مميزة « الجهاز النشيط » (أو إذا كان محرك قرص ، فإنه يكون محركاً نشيطاً) . وعندما يتداول ملف تم تخزينه على جهاز نشيط فإنه لا حاجة إلى تحديد الجهاز المميز . ومع كل .. فعندما يتداول ملف على جهاز آخر لابد من اعتبار الجهاز المميز كجزء من توصيف الملف .

مثال ١٠ ــ ٢

حاسب دقيق له محركان للأقراص المرنة مميزان كمحرك A ومحرك B. نفرض أن محرك A هو المحرك النشيط حالياً ، فإذا أردنا أن نشير إلى برنامج بيسك على المحرك A يسمى SAMPLE فيمكننا كتابة مواصفات الملف كالآتى :ـــ

SAMPLE.BAS

ومن ناحية أخرى إذا أردنا الإشارة إلى برنامج بيسك على المحرك B يسمى DEMO ، فلابد لنا من كتابة المواصفات الكلية للملف كالآتى :-

B:DEMO.BAS

ويمكن بسهولة للملف (مثل برنامج) مخزن على أحد الأجهزة أن يتداول ملفاً آخر (مثل ملف بيانات) مخزناً على جهاز آخر . ولعمل ذلك يمكننا ببساطة جعل تمييز الجهاز كجزء من تمييز الملف عند تداول الملف الثانى .

مثال ١٠ ــ ٣

نفترض مرة أخرى حاسباً دقيقاً له محركان للأقراص المرنه A و B حيث المحرك A نشيط . ولنفرض إننا نقوم بتشغيل برنامج بيسك مخزن على المحرك A وهذا البرنامج يتداول ملف بيانات توصل عشوائى على المحرك B يسمى STUDENTS.DAT . وعلينا إذاً الإشارة إلى ملف بيانات التوصل العشوائى كالآتى :-

B:STUDENTS.DAT

من داخل البرنامج

مثال ١٠ ــ ٤

شكل ١٠ ــ ١ يبين تغييراً فى البرنامج الذى سبقت كتابته فى مثال ٩ ــ ٣٠ لتشغيل درجات امتحانات الطلبة مستخدمين معلومات مخزنة على ملفات بيانات متتابعة البرنامج الأصلى مبين فى شكل ٩ – ١٠ . ومع كل فإن ملفات البيانات المتتابعة متداولة من محرك B وكل منها له امتداد DAT . متصل باسم الملف . السطور 40 ، 50 ، 250 و 260 تعكس هذه التغيرات فى الاستخدام .

```
10 REM ****** PROGRAM TO PROCESS STUDENT EXAMINATION SCORES ******
20 REM ******            USING SEQUENTIAL DATA FILES          ******
30 DIM C(15)
40 OPEN "I",1,"B:SCORES.DAT"
50 OPEN "O",2,"B:UPDATE.DAT"
60 INPUT #1,TITLE$
70 INPUT #1,TERM$
80 PRINT "Course title: ";TITLE$,"Term: ";TERM$
90 PRINT #2,TITLE$: PRINT #2,TERM$
100 PRINT: INPUT "Exam number";K
110 PRINT: INPUT "Calculate averages (Y/N) ";ANS$
120 PRINT: INPUT #1,N$
130 FOR I=1 TO K-1: INPUT #1,C(I): NEXT I
140 PRINT N$,
150 INPUT "New score";C(K)
160 PRINT #2,N$
170 SUM=0
180 FOR I=1 TO K: PRINT #2,C(I);: SUM=SUM+C(I): NEXT I
190 IF ANS$="N" OR ANS$="n" THEN 230
200 AVG=SUM/K
210 PRINT "Average=";AVG
220 PRINT #2,AVG
230 IF NOT EOF(1) THEN 120
240 CLOSE
250 KILL "B:SCORES.DAT"
260 NAME "B:UPDATE.DAT" AS "B:SCORES.DAT"
270 END
```

شكل ١٠ – ١

وعند تداول برنامج بيسك ، فإنه يمكن حذف كل من تمييز الجهاز وامتداد الملف بشروط معينة . وسنعرض المزيد حول خصائص ملف البيسك فى القسم التالى .

١٠ ــ ٢ أوامر نظام الحاسب الدقيق MICROCOMPUTER SYSTEM COMMANDS

تحتوى العديد من نسخ بيسك الحاسب الدقيق على أوامر معينة للنظام والتى لا توجد فى النسخ التقليدية للغة . ومن ناحية أخرى فإن بعض أوامر نظام تقليدى معين ليست متاحة على الحاسب الدقيق ، أو يتم تنفيذها بشكل مختلف وعلى ذلك فسنعطى بعض الاعتبارات لأوامر نظام الحاسب الدقيق الأكثر شيوعاً كما يتم تنفيذها فى ميكروسوفت بيسك .

وفى بعض الحاسبات الدقيقة يظهر البيسك أتوماتيكيا بمجرد تشغيلها . ومع كل فإنه من المعتاد فى الغالب أن يتداول الحاسب الدقيق نظام تشغيله بعد البدء فى التشغيل . ولكى ندخل البيسك من نظام التشغيل ، فلابد للمستخدم أن يطبع أمر BASIC. (هناك أوامر مشابهة متاحة لتوصيل لغات أخرى أو تطبيقات حاسبات دقيقة أخرى مثل معالجة الكلمات).

وبمجرد تداول البيسك ، فإن المستخدم سيرى رسالة مثل التى تظهر كالآتى :ــ

```
The IBM Personal Computer Basic
Version D2.10 Copyright IBM Corp. 1981, 1982, 1983
61327 Bytes free

Ok
```

وبعد ذلك فإنه يمكن للمستخدم تحميل برنامج البيسك وتشغيله وإظهار قائمة ....الخ على الشاشة وذلك بإدخال أوامر النظام المناسبة.. وسنلخص فيما يلى أوامر النظام الأكثر شيوعا (كما يتم تنفيذه فى ميكروسوفت بيسك لحاسب IBM الشخصى) .

الأمــــر	الغـــرض	أمثلـــة
AUTO	يرقم أسطر البرنامج المتتالية أوتوماتيكياً (والأرقام سوف تكون 30,20,10... إلا إذا تم توصيفها بغير ذلك) .	AUTO AUTO 100,5 (تبدأ بالسطر رقم 100 وتزاد بخطوة قدرها 5)
CLEAR	يجعل كل المتغيرات الرقمية تساوى صفراً ، وكل المتغيرات لسلسلة الحروف خالية .	CLEAR
CONT	يستأنف التنفيذ بعد إيقافه (مثل جملة STOP أو جملة END) .	CONT
DELETE	يحذف كتلة من السطور من برنامج موجود حالياً فى الذاكرة	DELETE 50 - 80 تحذف السطور من 50 الى 80
EDIT	يدخل أسلوب تنقيح السطر (انظر قسم ١٠ ــ ٧)	EDIT 100 (لينقح السطر 100)
FILES	يعرض قائمة بكل الملفات التى تم تخزينها على وحدة تخزين كبيرة (مثل قرص مرن)	FILES FILES "B:"
KILL	يحذف ملفًا من وحدة التخزين (مثل قرص مرن)	KILL "SAMPLE" KILL "B:STUDENTS.DAT"
LIST	يعرض قائمة بكل أو بجزء من البرنامج المخزن حالياً فى الذاكرة	LIST LIST 100—200 (اعرض قائمة بالسطور من 100 إلى 200)
LLIST	يعرض (يطبع) قائمة بكل أو بجزء من البرنامج المخزن حالياً فى الذاكرة وذلك على طابعة السطور	LLIST LLIST 100—200 (اعرض قائمة بالسطور من 100 إلى 200)

الأمـــــر	الغـــــرض	أمثلـــة
LOAD	يسترجع ملفاً من على وحدة التخزين (مثل قرص مرن) ويخزنه فى ذاكرة الحاسب .	LOAD "DEMO" (DEMO هو اسم الملف) LOAD "B: SAMPLE. BAS"
MERGE	يدمج البرنامج المخزن على وحدة التخزين الكبيرة (مثل قرص مرن) مع البرنامج الموجودة حالياً فى الذاكرة .	MERGE "TRIAL" MERGE "B:REPORT.BAS"
NAME	يعيد تسمية ملف مخزن على وحدة تخزين كبيرة (مثل قرص مرن)	NAME "SAMPLE" AS "NEW PROG" NAME "B: DEMO" AS "B: INVADERS.BAS"
NEW	يحذف البرنامج المخزن حالياً فى الذاكرة	NEW
RENUM	يعيد ترقيم سطور البرنامج أوتوماتيكياً (الأعداد ستكون 10 ، 20 ، 30 ،... إلا إذا تم تخصيص غير ذلك) .	RENUM RENUM 10,100,5 (ابدأ بالسطر الأصلى 10 وغيره للسطر 100 بزيادة قدرها 5)
RUN	ينفذ البرنامج الموجود حالياً فى الذاكرة أو يحمل برنامجاً ثم يقوم بتنفيذه .	RUN RUN "B:SAMPLE"
SAVE	يخزن البرنامج الموجود حالياً فى الذاكرة على وحدة تخزين كبيرة (مثل القرص المرن) .	SAVE "DEMO" SAVE "B:SAMPLE"
SYSTEM	يخرج من البيسك إلى نظام التشغيل	SYSTEM
TRON	يساعد فى تصحيح الخطأ ويسبب عرض أرقام سطور البرنامج عند تنفيذها .	TRON
TROFF	يلغى أمر TRON	TROFF

لاحظ أن الملفات المميزة المطلوبة محصورة بين علامات التنصيص (quotes) . وهذه خاصية لميكروسوفت بيسك . ولابد أن يكون مفهوماً أن التفاصيل التركيبية قد تختلف فى النسخ الأخرى من بيسك الحاسب الدقيق . هذا بالإضافة إلى أن الأوامر نفسها قد تختلف فى النسخ الأخرى من اللغة .

مثال ١٠ ــ ٥ تحديد الوقت من اليوم **Time of Day**

مثال ١٠ ــ ٢ يوضح جلسة نموذجية تحتوى على إستخدام البيسك على حاسب IBM الشخصى . وفى هذه الجلسة نرى ماذا يحدث عندما نبدأ بتشغيل الجهاز ويتم تداول البيسك . ونبدأ بتحميل برنامج يسمى DEMO.BAS والمخزن على القرص المرن النشيط (محرك A) ، ثم يتم عرض قائمة البرنامج وتشغيله . وبعد ذلك نقوم بتغيير البرنامج وذلك بإضافة جملتين PRINT إضافيتين (السطرين 35 ، 65) لعرض قائمة النسخة الجديدة ، ونقوم بتشغيلها ثم نحفظ النسخة الجديدة على المحرك النشيط . وفى النهاية نخرج من البيسك ونعود إلى نظام التشغيل .

وعند البداية فإن البرنامج نفسه يقرأ اسماً مثل "Sharon" ثم يحدد الوقت الحالى من اليوم (لاحظ أنه يتم تسجيل الوقت عند بدء تشغيل الحاسب) . ويقوم البرنامج عند ذلك بطبع الجمل الملائمة مثل "Good morning, Sharon" أو "Good afternoon, Sharon" أو "Good evening, Sharon" وفى هذه الجلسة بالذات ، فإن البرنامج يطبع "Good evening, Sharon" حيث أننا أدخلنا الساعة 19:36 (أى الساعة 7.36 مساءاً) عند بداية الجلسة . (لاحظ أن البرنامج يستخدم عدة دوال مكتبية جديدة ، كما هو مبين بقسم ٩ ــ ٣) .

ولاحظ أن البرنامج (الملف) المميز لا يحتاج إلى تضمين مواصفات المحرك حيث أنه يتم تحميل وحفظ البرنامج فى المحرك النشيط حالياً . ولاحظ أيضاً أن الامتداد مفروض أن يكون BAS . وحيث أن البرنامج الحالى ليس له هذا الامتداد ، فإننا لن نحتاج إلى تضمينه فى الملف المميز .

وكما فى الأمثلة السابقة من هذا النوع ، تكون استجابة المستخدم تحتها خط .

وتذكر أن هذا المثال هو تمثيل لجلسة برمجة حاسب دقيق بميكروسوفت بيسك . وقد تختلف التفاصيل من حاسب إلى آخر . هذا بالإضافة إلى أن الجلسة قد تختلف كلية مع بعض نسخ أخرى من بيسك الحاسب الدقيق .

ولابد للقارىء أن يرجع إلى ملحق (د) لملخص أكثر شمولاً لميكروسوفت بيسك يحتوى على أوامر النظام الأكثر استخداماً .

```
Current date is Tue  1-01-1980
Enter new date:  9-5-1985
Current time is  0:00:12.74
Enter new time:  19:36

The IBM Personal Computer DOS
Version 2.10 (C)Copyright IBM Corp 1981, 1982, 1983

A>BASIC

The IBM Personal Computer Basic
Version D2.10 Copyright IBM Corp. 1981, 1982, 1983
61327 Bytes free

Ok

LOAD "DEMO"
Ok

LIST
10 REM *** "Good morning" program ***
20
30 INPUT "Hi, what's your name? ",N$
40 HOUR = VAL(LEFT$(TIME$,2))
50 IF HOUR < 12 THEN PRINT "Good morning, "; ELSE IF HOUR < 18 THEN
   PRINT "Good afternoon, "; ELSE PRINT "Good evening, ";
60 PRINT N$
70 END
Ok

RUN
Hi, what's your name? Sharon
Good evening, Sharon
Ok

35 PRINT
65 PRINT

LIST
10 REM *** "Good morning" program ***
20
30 INPUT "Hi, what's your name? ",N$
35 PRINT
40 HOUR = VAL(LEFT$(TIME$,2))
50 IF HOUR < 12 THEN PRINT "Good morning, "  ELSE IF HOUR < 18 THEN
   PRINT "Good afternoon, "; ELSE PRINT "Good evening, ";
60 PRINT N$
65 PRINT
70 END
Ok

RUN
Hi, what's your name? Sharon

Good evening, Sharon

Ok

SAVE "DEMO"
Ok

SYSTEM

A>
```

شكل ١٠ ــ ٢

١٠ ـــ ٣ شاشة عرض TV   THE TV MONITOR

معظم الحاسبات الدقيقة وكثير من النهايات الطرفية للمشاركة الزمنية تستخدم شاشة عرض TV (أى وحدة CRT) كجهاز مخرجات أساسى . وتعرض هذه الوحدات فى المعتاد 24 أو 25 سطراً من النص ، وبعدد حروف يصل إلى 80 حرفاً فى السطر (ومع ذلك فيجب أن نلاحظ أن بعض الحاسبات الدقيقة تولد مخرجات ذات حروف كبيرة ، ويترتب على ذلك عدد حروف أقل فى السطر وربما أيضاً سطور أقل) . ويمكن أيضاً توليد العرض البيانى على هذه الأجهزة (انظر فصل ١٢) . وكل من نوعى شاشة العرض ذات اللون الواحد والمتعددة الألوان شائعة الاستخدام حالياً .

مسح الشاشة   Clearing the Screen

ولشاشات عرض TV خصائص فريدة معينة لابد من أخذها فى الاعتبار عند كتابة برامج البيسك . فمثلاً .. يمكن لشاشة عرض TV أن تعرض عدداً محدوداً فقط من السطور فى وقت واحد (وذلك بمقارنتها بطابع السطور الذى يمكنه أن يستمر طالما كان هناك ورق كاف) . وعلى ذلك فلابد أن نمسح الشاشة دورياً حتى يمكن أن يبدأ عرض جديد فى أعلى الركن الأيسر من شاشة خالية .

وعلى حاسب IBM الشخصى تستخدم الجملة CLS (مسح الشاشة) لمسح الشاشة . وهذا الأمر يمسح كل ما كان معروضاً على الشاشة ثم يحرك المؤشر إلى أعلى الركن الأيسر .

تحريك المؤشر   Moving the Cursor

توجد خاصية أخرى فريدة لشاشة عرض TV وهى المؤشر ، وهذه رمز غماز ، عبارة عن مستطيل صغير أو سطر أفقى ، ويستخدم ليبين موضعاً محدداً على الشاشة (مثل سطر أو عمود معين) . والمؤشر له عدة أغراض مفيدة ، فمثلاً يمكن استخدامه لجذب الانتباه لمكان معين على الشاشة (لقراءة رسالة مثلاً) أو طلب بيانات مدخلة أو لبيان مكان ظهور رسالة المخرجات التالية .

وعند تصحيح ملف (مثلاً برنامج البيسك) فإن موضع المؤشر تتحكم فيه لوحة المفاتيح . وتحتوى معظم لوحات مفاتيح الحاسب الدقيق على مفاتيح خاصة تستخدم لتحريك المؤشر إلى أعلى أو إلى أسفل أو على الجانبين (وسنتحدث عن ذلك بإفاضة فى القسم التالى) . ومع كل ، فعند تنفيذ برنامج فإن تحديد موضع المؤشر يتم داخل البرنامج . والقدرة على تحريك المؤشر بهذه الطريقة مهمة على وجه التحديد عند كتابة برامج تفاعلية (نمط تحاورى) . وعلى ذلك فإن معظم نسخ البيسك للحاسب الدقيق تحتوى على جمل خاصة لتحديد موضع المؤشر تحت تحكم البرنامج .

وعند تنفيذ برنامج ميكروسوفت بيسك على حاسب IBM الشخصى ، فإن موضع المؤشر يتم تحديده بجملة LOCATE . ويمكن أن تظهر جملة LOCATE نموذجية كالآتى :-

```
LOCATE 5,20
```

وهذه الجملة الخاصة تتسبب فى تحديد وضع المؤشر على الصف 5 (أى الصف الخامس من أعلى) والعمود 20 (من اليسار) . ويمكن ببساطة تحديد مواضع الصفوف والأعمدة الأخرى بتغيير المعالم الرقمية فى هذه الجملة .

ومن المهم أن نعرف أنه يمكن تحريك المؤشر فى أى مكان على الشاشة بهذه الطريقة بدون مسح أى جزء من النص المكتوب سابقاً . وعلى ذلك ..فإنه يمكن تحريك المؤشر إلى كلمة معينة أو رمز معين موجود فعلاً على الشاشة .

مثال ١٠ ــ ٦

إذا أردنا مسح الشاشة ثم وضع المؤشر فى مركز الشاشة (أى الصف 13 والعمود 40) على حاسب IBM الشخصى ، فإننا نكتب .

```
10 CLS
20 LOCATE 13,40
30 ANS$ = INPUT$(1)
```

لاحظ أن الجملة الأخيرة تسبب وقفة فى تنفيذ البرنامج ، وبذلك تسمح لرؤية المؤشر فى موضعه الجديد . ويستأنف تنفيذ البرنامج إذا ما تم الضغط على أى مفتاح .

ولا تحتوى بعض نسخ ميكروسوفت بيسك القديمة على أى جمل لتحريك المؤشر صراحة . وفى هذه الحالات يمكن محاكاة جملة CLS بكتابة PRINT CHR$(12), حيث 12 هى الرقم (العشرى) لكود ASCII لتغذية سطر . ويمكن محاكاة جملة LOCATE أيضاً بطباعة مجموعة من السطور الخالية المتتالية ويتبعها عدة أماكن خالية فى السطر المطلوب .

مثال ١٠ ــ ٧

حتى يمكن مسح الشاشة ووضع المؤشر فى مركزها (أى صف 13 عمود 40) باستخدام حاسب دقيق من النوع القديم بدون أوامر حركة المؤشر صراحة ، يمكننا كتابة

```
10 PRINT CHR$(12)
20 FOR ROW = 1 TO 12 : PRINT : NEXT ROW
30 FOR COL = 1 TO 40 : PRINT " "; : NEXT COL
40 INPUT " ",A$
```

(قارن مع مثال ١٠ ــ ٦)

هذه الطريقة ، بالطبع ، أقل ملاءمة من المعروضة فى مثال ١٠ ــ ٦ . ومن ثم فيجب ألا تستخدم إلا إذا كانت هى الطريقة الوحيدة المتاحة .

توجد ثلاث دوال تستخدم مع جملة PRINT من أجل وضع المؤشر على السطر . هى (TAB) ، SPC و SPACE$. أولها (TAB) تستخدم لوضع المؤشر فى بداية السطر نسبياً كما نوقش فى الفصل الخامس . والدالة الثانية (SPC) تسبب تحريك المؤشر عدداً معيناً من الفراغات وراء نطاق آخر وضع له (وهو عادة مجاور لآخر حرف مطبوع) . والدالة الأخيرة (SPACE$) تشبه SPC حيث أنها تعيد سلسلة حروف تحتوى على عدد مميز من الأماكن الخالية .

مثال ١٠ ــ ٨

```
10 PRINT "RED";TAB(6);"BLUE"
```

سوف تتسبب فى طباعة السلسلة الحرفية BLUE ابتداء من العمود 6 (وهو حقيقة العمود السابع فى السطر) بينما الجملة :

```
20 PRINT "RED";SPC(6);"BLUE"
```

سوف تتسبب فى ظهور 6 مسافات خالية بين BLUE, RED (ومن ثم فسوف تبدأ BLUE فى العمود رقم 9 ، وهو حقيقة العمود العاشر فى السطر) .

ويمكن كتابة جملة **PRINT** الأخيرة كالآتى :

```
20 PRINT "RED";SPACE$(6);"BLUE"
```

سوف تعيد دالة **POS** المكان الحالى للمؤشر فى سطر ما . وهذا يسمح بتفرع مشروط مُقَاس إلى مكان المؤشر كما هو موضح فيما بعد . (لاحظ أن هذه الدالة تهمل القيمة الموصوفة للخلاصة المطلوبة) .

مثال ١٠ ــ ٩

ادرس الجملتين التاليتين :

```
10 PRINT NAME$;
20 IF POS(1) < 20 THEN PRINT TAB(20); ADDRESS$
   ELSE PRINT SPC(2); ADDRESS$
```

سوف تبدأ السلسلة الحرفية الثانية ADDRESS $ فى العمود رقم 20 (حقيقة العمود رقم 21) إذا كان طول سلسلة الحروف الأولى (NAME$) أقل طولا من 20 حرفاً ، وإلا فسوف تبدأ سلسلة الحروف الثانية بعد ثلاثة أعمدة من نهاية سلسلة الحروف الأولى (سوف يفصل السلسلتين الحرفيتين مكانان خاليان) .

اللفة الرأسية Vertical Scrolling

خاصية أخرى من خواص شاشة عرض TV هى خاصية اللفة الرأسية . وهذا يشير إلى الحركة الرأسية للكتابة المعروضة على الشاشة عند إدخال سطر جديد من النص أو عرضه على شاشة وهى ممتلئة تماما . وعند حدوث ذلك ، فإن السطر الذى كان فى أعلى الشاشة سابقاً سوف يفقد ، وكل السطور الباقية سوف تتحرك إلى أعلى بمقدار مكان سطر والسطر الجديد يظهر عند قاع الشاشة .

يمكن أن تستخدم اللفة الرأسية فى بعض الأحيان على نحو مفيد لإعطاء انطباعات خاصة ، كما هو موضح فى مثال ١٠ — ١٠ فيما بعد . ويمكن أيضاً أن يكون مزعجاً ، حيث أنه يمكن أن يتسبب فى اختفاء نص معروض قبل إتمام قراءته ، ولكن هذه المشكلة يمكن عادة حذفها (أو تقليلها على الأقل) باستخدام الطرق الفنية الخاصة للبرمجة مثل حلقات تكرارية فارغة FOR –TO وذلك لتوليد تأخير فى الوقت ، واستخدام دوال INPUT INKEY$ لإنشاء وقفات . انظر مثال ٩ — ٢٢، أو إعادة وضع المؤشر رأسياً أو استخدام آلة الطباعة لعرض النص الطويل .

ومشكلة تتعلق بذلك عن قرب هى مسح الشاشة (بواسطة جمل CLS) قبل أن نتمكن من قراءة رسالة معروضة . ويمكن أيضاً التغلب على هذه المشكلة من خلال استخدام حلقات تكرارية فارغة **FOR-TO** أو دوال **INKEY$** أو **INPUT$**

مثال ١٠ — ١٠ برمجة شاشة عرض TV (لا شئ يمكن أن يفضى إلى خطأ ، يفضى إلى خطأ ..)

Programming a TV Display (Nothing Can Go Wrong, Go Wrong, Go Wrong, . . .)

نرى فى شكل ١٠ — ٣ برنامج حاسب دقيق قصير للتسلية مكتوب بميكروسوفت بيسك لحاسب IBM الشخصى وهو يوضح استخدام تأخير الوقت ، وحركة المؤشر واللفة الرأسية المعتمدة . وقد صمم البرنامج ليتسبب فى ملء شاشة ذات 80 حرفاً بهذه الرسالة

```
GO WRONG! GO WRONG! GO WRONG! GO WRONG! GO WRONG! GO WRONG! GO WRONG!
```

لكى تملأ شاشة 80 حرف ويلف رأسياً وبذلك يخلق الانطباع الكاذب بأن الحاسب الدقيق والذى تثق فيه والمفروض إلا يخطىء قد فعل ذلك حقيقة .

وهذا البرنامج يحتوى على عدد من أوامر CLS (السطور 30 ,80 ,200) والعديد من أوامر LOCATE (السطور ,100 ,130 ,200 ,220 ,240 ,30 ,50 ,08) .

وتسمح هذه الأوامر لمخطط البرامج أن يتحكم فى مظهر الشاشة وخصوصاً مواضع طباعة الرسائل المتعددة . ويحتوى البرنامج أيضاً على حلقتين تكراريتين فارغتين FOR- TO (السطران 70 ,120) وهما اللتين تنشئان تأخير فى الوقت . ويكون التأثير النهائى هو عرض ديناميكى كرسوم متحركة بتأثير مرئى قوى .

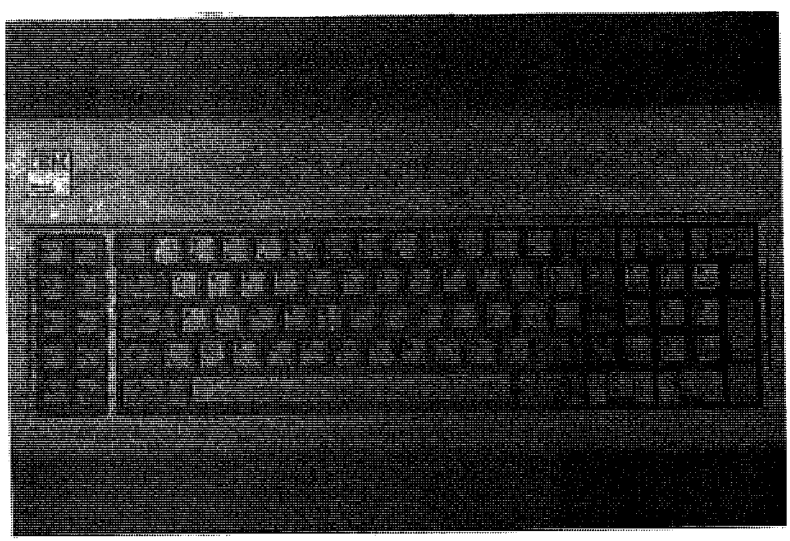
ونشجع القارىء أن يُجرى هذا البرنامج حتى يشاهد الملف الرأسى الحقيقى . وهذا يزيد كثيراً من تفهم القارىء للبرنامج . وإنه لمن المشوق أيضاً أن يقوم بتنفيذ البرنامج على آلة طباعة أو وحدة طرفية ويقارن أثرها مع عرض شاشة TV .

```
10  REM  *** TV-DISPLAY DEMO (IBM Version 1) ***
20
30  CLS : LOCATE 11,28
40  PRINT "WELCOME TO THE WONDERFUL"
50  LOCATE 14,28
60  PRINT "WORLD OF MICROCOMPUTERS!"
70  FOR I=1 TO 3000 : NEXT I
80  CLS: LOCATE 11,18
90  PRINT "Just relax, enjoy yourself and remember that"
100  LOCATE 14,25
110  PRINT "NOTHING CAN POSSIBLY GO WRONG!"
120  FOR I = 1 TO 3000 : NEXT I
130  LOCATE 24
140  FOR I = 1 TO 24
150    FOR J = 1 TO 8
160       PRINT "GO WRONG! ";
170    NEXT J
180    PRINT : PRINT
190  NEXT I
200  CLS: LOCATE 11,37
210  PRINT "DARN!"
220  LOCATE 15,30
230  PRINT "Something went wrong!"
240  LOCATE 23
250  END
```

شكل ١٠ ــ ٣

١٠ ــ ٤ لوحة المفاتيح THE KEYBOARD

تحتوى معظم لوحات مفاتيح الحاسبات الدقيقة على عدد مفاتيح أكبر كثيراً من مفاتيح الآلة الكاتبة العادية. ومن المعتاد وجود مجموعتين أوثلاث إضافية من المفاتيح . وهذه المفاتيح الإضافية قد تحتوى على مجموعة من مفاتيح المؤشر وحركة الشاشة ، ولوحة مفاتيح صغيرة رقمية (أى مجموعة من المفاتيح الرقمية التى تشابه المفاتيح الرقمية للآلة الكاتبة إلا أن لها تنظيماً معيناً يشابه نوع الآلة الحاسبة) ومجموعة من مفاتيح خاصة للدالة . ويمكن برمجة عدد معين من هذه المفاتيح لتقوم بأعمال خاصة خلال تنفيذ برامج بيسك .



شكل ١٠ ــ ٤

اعتبر مثلاً لوحة المفاتيح المستخدمة مع حاسب IBM الشخصى المبين فى شكل ١٠ ــ ٤ . ولاحظ أن مفاتيح الآلة الكاتبة موجودة فى مركز لوحة المفاتيح . وعلى اليمين توجد مجموعة من المفاتيح للقيام بعمليتين مختلفتين : وهى تستخدم عادة كمفاتيح للمؤشر ولحركة الشاشة . رغم أنه يمكن أن تصبح مفاتيح عددية إذا كان مفتاح « ''NumLock'' » (الموجود فى أعلى الركن الأيمن) قد تم الضغط عليه . وتحتوى الجهة اليسرى من لوحة المفاتيح على مجموعة من مفاتيح الدالة معنونة F1 حتى F10 .

وتحتوى النسخة المتقدمة من ميكروسوفت بيسك التى تصاحب حاسب IBM الشخصى (تسمى BASICA) على العديد من الأوامر الخاصة التى تسمح للحاسب بتحديد إذا ما كانت أى من مفاتيح الدالة أو مفاتيح معينة لتحريك الشاشة قد تم الضغط عليه* . يمكن بعد ذلك أن يستجيب الحاسب لأى طريقة قد يراها المبرمج من خلال استدعاء برامج فرعية خاصة (أى من خلال شكل معين من جملة ON— GOSUB) وسنرى كيف يتم ذلك فى المثال التالى .

مثال ١٠ ــ ١١ برمجة مفاتيح الدالة Programming the Function Keys

يمثل شكل ١٠ـــ ٥ برنامج بيسك بسيطاً مكتوباً بواسطة ميكروسوفت بيسك المتقدم (BASICA) لحاسب IBM الشخصى . وهذا البرنامج يحتوى على حلقة تكرارية تعرض الأرقام الصحيحة الموجبة المتتالية 3,2,1...32767 على الشاشة مع رقم واحد صحيح على كل سطر . وحيث أن الشاشة يمكنها أن تعرض 24 سطراً فى وقت واحد فإن الأرقام تتحرك رأسياً إلى أعلى بمجرد أن تمتلىء الشاشة .

والبرنامج يحتوى على امكانية انهاء الحسابات فى أى وقت يريده المستخدم ببساطة وذلك بالضغط على مفتاح الدالة F2 . كما يمكن ايضا عكس الوان الشاشة (أى تغير الحروف البيضاء على خلفية سوداء إلى حروف سوداء على خلفية بيضاء والعكس بالعكس) فى أى وقت وذلك بالضغط على مفتاح الدالة F1 أثناء تنفيذ البرنامج .

```
10 REM *** FUNCTION KEY DEMO ***
20 '
30 ON KEY(1) GOSUB 160
40 ON KEY(2) GOSUB 200
50 KEY(1) ON : KEY(2) ON
60 '
70 CLS : FLAG = 1
80 PRINT "FUNCTION KEY DEMONSTRATION" : PRINT
90 PRINT "Press F1 to REVERSE the screen, F2 to STOP" : PRINT
100 PRINT "Press any other key to begin"
110 DUMMY$ = INPUT$(1)
120 FOR COUNT = 1 TO 32767 : PRINT COUNT : NEXT COUNT
130 END
140 '
150 REM *** SUBROUTINE TO REVERSE THE SCREEN ***
160 IF FLAG = 1 THEN COLOR 0,7 : FLAG = 0 ELSE COLOR 7,0 : FLAG = 1
170 RETURN
180 '
190 REM *** SUBROUTINE TO TERMINATE THE COMPUTATION ***
200 END
```

شكل ١٠ ــ ٥

دعنا الآن نختبر هذا البرنامج بالتفصيل . السطران 40,30 يصاحبان كل مفتاح دالة ببرنامج فرعى معين ، والسطر 50 ينشط هذه المصاحبة . السطر 70 يمسح الشاشة ثم يخصص قيمة 1 إلى FLAG مشيرا إلى الحروف البيضاء على خلفية سوداء ، ثم تظهر بعد ذلك عبارة مبدئية فى السطر 80 حتى 10. وتولد الأرقام الصحيحة المتتالية وتطبع فى الحلقة التكرارية FOR-TO والتى تظهر على السطر 120 ، والسطر 130 يمثل نهاية الجزء الرئيسى للبرنامج .

* فى الحقيقة يدعم حاسب IBM الشخصى ثلاثة أنواع من نسخ ميكروسوفت بيسك ، تسمى كاسيت بيسك Cassette BASIC، ديسك بيسك Disk BASIC وبيسك المتقدم Advanced BASIC على الترتيب .

والسطور 150 إلى 170 . تكون البرنامج الفرعى الأول (مصاحبا F1) والذى يعكس الألوان . والجزء الرئيسى من البرنامج الفرعى هو جملة IF-THEN-ELSE على السطر 160 . ويتم عكس الألوان بجملة COLOR . وعلى ذلك فإذا كانت FLAG=1 مشيرة إلى أبيض على أسود ، فإن الألوان تعكس بتحديد COLOR 0,7 (0 يشير إلى الحروف السوداء و7 تشير إلى الخلفية البيضاء) . وبعد ذلك يتم وضع FLAG=0 . وبالمثل فإذا كانت FLAG لا تساوى 1 فإن الألوان تعكس بتحديد COLOR 7,0 (حروف بيضاء وخلفية سوداء) . ثم تعود FLAG إلى 1 . وجملة RETURN فى سطر 170 تسبب التحكم فى الرجوع للجملة التى تلى الموضع الذى تم فيه الضغط على مفتاح الدالة .

ويحتوى البرنامج الفرعى الثانى ببساطة على ملاحظة تحديدية وجملة END . وليس المطلوب هنا جملة RETURN حيث أن الغرض من هذا البرنامج الفرعى هو إنهاء الحسابات .

وسنتحدث بإفاضة عن جملة COLOR فيما بعد فى هذا الفصل (انظر قسم ١٠ ــ ٦) . ومع كل فلابد أن يظهر لنا أن هذه الجملة تستخدم لإنتاج نص ملون على خلفية من الألوان المختلفة،ويمكن أيضاً أن تستخدم لتوليد بيانيات ملونة كما سنرى فى الفصل الثانى عشر .

وفى النهاية فإنه على القارىء أن يرى هذا البرنامج أثناء تشغيله حتى يمكنه استيعاب كيفية عمله وما يقوم به . وإذا كان حاسب IBM الشخصى متاحاً ، فإننا نشجع القارىء بشدة أن يقوم بإدخال البرنامج وتشغيله ضاغطاً على FI عدة مرات أثناء تنفيذ البرنامج ، ثم الضغط على F2 لإيقاف تنفيذ البرنامج .

١٠ ــ ٥ وحدات إدخال أخرى قابلة للبرمجة

OTHER PROGRAMMABLE INPUT DEVICES

يمكن للحاسبات الدقيقة أن تستخدم العديد من الأنواع المختلفة لأجهزة الإدخال القابلة للبرمجة بالإضافة إلى لوحة المفاتيح . وهناك نوعان منها يشيع استخدامهما ، وهما القلم الضوئى وعصا التوجيه . والقلم الضوئى هو جهاز للإشارة يمكن برمجته للتعرف على موقع ما على الشاشة ، ثم يقوم بتنشيط دالة ما . وبالمثل فإن عصا التوجيه هى جهاز لتحديد موضع يمكن برمجته لتحديد موضع المؤشر ثم يقوم بتنشيط دالة بالضغط على زر . ويسمح ميكروسوفت بيسك ببرمجة هذه الأجهزة بما يشابه طريقة مفاتيح داله لوحة المفاتيح كما هو موضح فى قسم ١٠ ــ ٤ .

اعتبر مثلاً القلم الضوئى الذى يستخدم كما هو موضح فى شكل ١٠ ــ ٦ . وهو جهاز حساس للضوء وبه مفتاح فى نهايته . وهذا المفتاح يتأثر بضغط القلم الضوئى على مصدر ضوئى على شاشة عرض TV . وموضع هذا المصدر الضوئى (صف وعمود) يمكن أن يتحدد بواسطة الحاسب .

ويحتوى ميكروسوفت بيسك المتقدم على جمل خاصة متعددة ودوال صممت لتستخدم مع القلم الضوئى . واستخدامات هذه الخصائص الخاصة موضحة فى المثال التالى .



شكل ١٠ ــ ٦

مثال ١٠ ــ ١٢ برمجة قلم ضوئى Programming a Light Pen

نرى فى شكل ١٠ ــ ٧ برنامج بيسك متقدم مكتوب لحاسب IBM الشخصى والذى يسمح للمستخدم أن يختار واحداً من عدة بنود معروضة على شكل قائمة على شاشة عرض TV . ويتم الاختيار بواسطة قلم ضوئى . وبمجرد اختيار أحد البنود يتم اتخاذ إجراء خاص بهذا الاختيار .

وسيقوم البرنامج على وجه الخصوص بتوليد قائمة تحتوى على أسماء ثمانى لغات مختلفة ، وتسبق كل منها كتلة مربعة من الضوء . ويقوم المستخدم بالضغط على طرف القلم الضوئى مقابلاً للكتلة التى تناظر اللغة التى يختارها . وبعد ذلك تظهر عبارة ترحيب باللغة المختارة بجوار اسم اللغة . وستظل عبارة الترحيب على الشاشة لمدة ثانية أو ثانيتين (وهى فترة طويلة تكفى لقراءتها بالراحة) ثم يتم مسحها . وبعد ذلك قد يختار المستخدم لغة أخرى وتتم إعادة العملية . ويستمر ذلك حتى يختار المستخدم آخر سطر فى القائمة (END) والذى يتسبب فى إيقاف البرنامج .

دعنا الآن نختبر البرنامج الحقيقى ببعض التفصيل . السطر 30 يصاحب إشارة تصدر من القلم الضوئى مع برنامج فرعى . والسطر 40 ينشط هذه المصاحبة . والسطر 50 يغرف سلسلة الحروف التى تحتوى على كتلة من الضوء . ويتم توليد القائمة بواسطة السطور 90 إلى 190 . والسطر 200 يدور فى حلقة تكرارية حول نفسه عدداً غير محدود من المرات مسبباً انتظار الحاسب لإشارة من القلم الضوئى .

```
10 REM *** LIGHT PEN DEMO ***
20 '
30 ON PEN GOSUB 240
40 PEN ON
50 SQUARE$=CHR$(219)+CHR$(219)
60 '
70 REM *** MAIN LOOP ***
80 '
90 CLS : PRINT "Multi-lingual greetings"
100 LOCATE 4,1 : PRINT "Please select a language"
110 LOCATE 6,4 : PRINT SQUARE$;SPC(4);"English"
120 LOCATE 8,4 : PRINT SQUARE$;SPC(4);"French"
130 LOCATE 10,4 : PRINT SQUARE$;SPC(4);"German"
140 LOCATE 12,4 : PRINT SQUARE$;SPC(4);"Hawaiian"
150 LOCATE 14,4 : PRINT SQUARE$;SPC(4);"Hebrew"
160 LOCATE 16,4 : PRINT SQUARE$;SPC(4);"Italian"
170 LOCATE 18,4 : PRINT SQUARE$;SPC(4);"Japanese"
180 LOCATE 20,4 : PRINT SQUARE$;SPC(4);"Spanish"
190 LOCATE 22,4 : PRINT SQUARE$;SPC(4);"END"
200 GOTO 200
210 '
220 REM *** SUBROUTINE TO PRINT 'HELLO' ***
230 '
240 ROW=PEN(6)
250 IF ROW=22 THEN LOCATE 23,1 : END
260 LOCATE ROW,20
270 IF ROW=6 THEN PRINT "Hello" : GOTO 350
280 IF ROW=8 THEN PRINT "Bonjour" : GOTO 350
290 IF ROW=10 THEN PRINT "Guten tag" : GOTO 350
300 IF ROW=12 THEN PRINT "Aloha" : GOTO 350
310 IF ROW=14 THEN PRINT "Shalom" : GOTO 350
320 IF ROW=16 THEN PRINT "Buon giorno" : GOTO 350
330 IF ROW=18 THEN PRINT "Konichihua" : GOTO 350
340 IF ROW=20 THEN PRINT "Buenos dias"
350 FOR COUNT=1 TO 1000 : NEXT COUNT
360 LOCATE ROW,20 : PRINT SPACE$(15)
370 RETURN
380 END
```

شكل ١٠ ــ ٧

وبمجرد اكتشاف إشارة من القلم الضوئى (نتجت من القلم الضوئى بالضغط على إحدى الكتل الضوئية مسبباً إطلاق مفتاحه) فإن تحكم البرنامج يقفز أتوماتيكياً إلى السطر 240 . وهذه الجملة تستخدم دالة PEN . ويتسبب المعامل 6 فى أن تعيد الدالة رقم الصف ، حيث تم تنشيط القلم الضوئى ، ثم يتم تخصيص هذه القيمة للمتغير ROW . وبعد ذلك يحرك السطر 260 المؤشر للاستعداد لعرض عبارة التحية المناسبة . والسطور 270 إلى 340 تختار عبارة التحية المناسبة وتتسبب فى عرضها . ويتم توليد تأخير الوقت بواسطة الحلقة التكرارية الفارغة FOR—TO فى سطر 350 . وفى النهاية يتم مسح التحية (تظهر مسافات خالية بدلا منها) فى سطر 360 ، ويعود التحكم إلى السطر 200 . وتستمر هذه العملية حتى يتم اختيار الصف 22 مسبباً إيقاف البرنامج (السطر 250) .

ويتسبب تنفيذ البرنامج فى ظهور قائمة على الشاشة مشابهة لتلك المبينة فى شكل ١٠ ــ ٨ . (لاحظ أن الكتل القائمة تظهر ككتل من الضوء على شاشة عرض TV) . ويقوم المستخدم بالضغط على القلم الضوئى مقابلاً إحدى الكتل لتظهر التحية المناسبة على يمين القائمة المختارة حالياً . وعلى سبيل المثال إذا ضغطنا على القلم الضوئى مقابلاً للكتلة الثانية ستظهر التحية "Bonjour" على يمين كلمة "French" . وستختفى التحية بعد تأخير قصير فى الوقت . ويمكن بعد ذلك أن يقوم المستخدم باختيار آخر . ولاحظ أنه يمكن إنهاء البرنامج فى أى وقت ببساطة وذلك باختيار الكتلة الأخيرة .

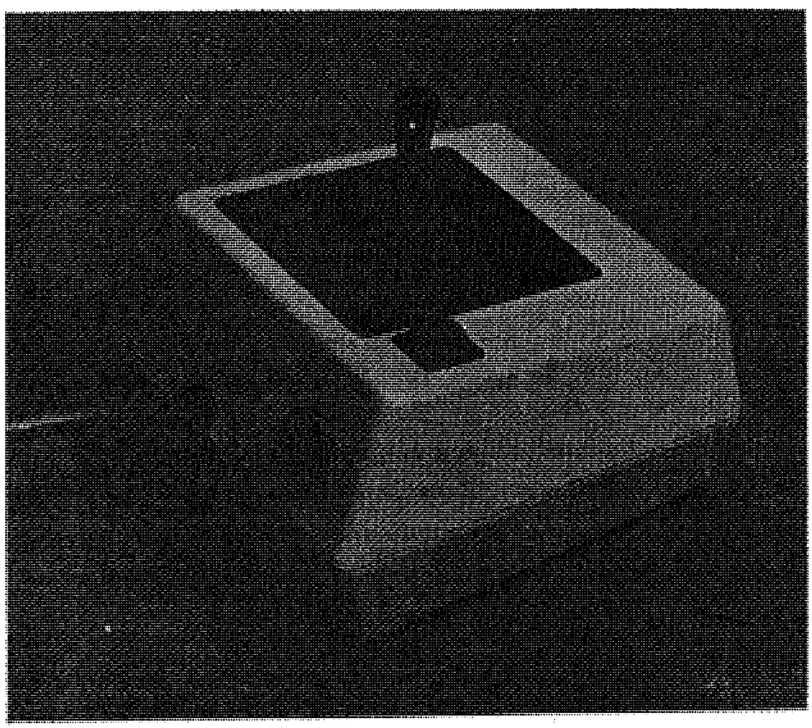
Multi-lingual greetings

Please select a language

- English
- French
- German
- Hawaiian
- Hebrew
- Italian
- Japanese
- Spanish
- END

شكل ١٠ ــ ٨

نفترض الآن عصا توجيه نموذجية كما هو مبين فى شكل ١٠ ــ ٩ . ولهذا الجهاز يد (أى عصا) يمكن تحريكها فى اتجاهين مختلفين (أعلى / أسفل ويسار/ يمين) . ويتم وصف موضع اليد بواسطة زوج من القيم الصحيحة تمثل الإحداثين x و y (أى أنه يمكن تمثيل كل إحداثى بواسطة رقم صحيح ينحصر بين 0 إلى 127) . وهذه القيم ترسل للحاسب باستمرار حيث يمكن تحويلها إلى موضع للمؤشر (أى رقمى الصف والعمود) . وإذا ما تحركت يد عصا التوجيه فإن موضع المؤشر يتغير نتيجة لذلك . وعلى ذلك يمكن برمجة الحاسب لتحريك المؤشر حول الشاشة نتيجة لحركة يد عصا التوجيه .



شكل ١٠ ــ ٩

ويوجد لعصا التوجيه زران (مفتاحان) يمكن استخدامهما لإرسال إشارات مفردة للحاسب . وعلى ذلك فإنه يمكن أن نضع المؤشر فى المكان المطلوب ، وذلك بتحريك يد عصا التوجيه ، ثم تنشيط دالة ما بالضغط على أحد الأزرار . ويمكن برمجة كل زر بمفرده لتنشيط دالته الفريدة .

ويحتوى ميكروسوفت بيسك المتقدم على العديد من الجمل والدوال الخاصة التى يمكن استخدامها لبرمجة عصا التوجيه . ويتم توضيح هذه الطريقة بالمثال التالى

مثال ١٠ ــ ١٣ برمجة عصا التوجيه Programming a Joystick

يحتوى شكل ١٠ ــ ١٠ على برنامج بيسك متقدم يسمح بتشغيل حاسب IBM الشخصى تحت تحكم عصا التوجيه . ويبدأ البرنامج بمسح الشاشة كلية وبذلك يمكن للمستخدم أن يحرك المؤشر إلى أى موضع مرغوب فيه، وذلك بتحريك يد عصا التوجيه. ويمكن للمستخدم عند أى موضع للمؤشر أن يعرض على الشاشة نجمة (*) ، وذلك بالضغط على الزر 1 (وهو الزر الأقرب لليد فى شكل ١٠ ــ ٩) . ويمكن أيضاً مسح أى نجمة موجودة بالضغط على الزر 2 .

```
10 REM *** JOYSTICK DEMO ***
20 '
30 CLS
40 ON STRIG(0) GOSUB 170
50 ON STRIG(4) GOSUB 200
60 STRIG(0) ON : STRIG(4) ON
70 '
80 REM *** MAIN LOOP ***
90 '
100 X=STICK(0) : Y=STICK(1)
110 ROW=23*(Y/127) + 1
120 COL=79*(X/127) + 1
130 LOCATE ROW,COL,1
140 GOTO 100
150 '
160 REM *** SUBROUTINE TO PRINT AN ASTERISK ***
170 PRINT "*"; : RETURN
180 '
190 REM *** SUBROUTINE TO PRINT A BLANK SPACE ***
200 PRINT " "; : RETURN
210 END
```

شكل ١٠ ــ ١٠

ويبدأ البرنامج بمسح الشاشة (سطر 30) . والسطران 40 و 50 يصاحبان إشارات تأتى من الزرين 1 و 2 مع البرامج الفرعية التى تطبع نجمة أو مكاناً خالياً على الترتيب ، ثم يقوم السطر 60 بتنشيط هذه المصاحبة .

السطور 100 إلى 140 تعرف حلقة تكرارية وهى التى تشكل قلب البرنامج . ويكتشف السطر 100 القيم الرقمية للمتغيرين X و Y ويتوقف ذلك على موضع يد عصا التوجيه . وستقع كل من هاتين القيمتين ما بين 0 ,127 . وعلى ذلك فإن النقطة Y=0, X=0 تشير إلى أعلى الركن الأيسر . والنقطة Y=127, X=127 تشير إلى أسفل الركن الأيمن ، ثم يقوم السطران 110 و120 بتحويل هذه القيم إلى رقم الصف (وتنحصر من 1 إلى 24) ورقم العمود (وينحصر من 1 إلى 80) . ويقوم السطر 130 بتحريك المؤشر إلى هذا الموقع الجديد . وفى النهاية ينهى السطر 140 الحلقة التكرارية (ويكتشف بذلك قيماً جديدة لكل من X و Y استجابة لحركة يد عصا التوجيه) ويعيد تحديد وضع المؤشر حسب ذلك .

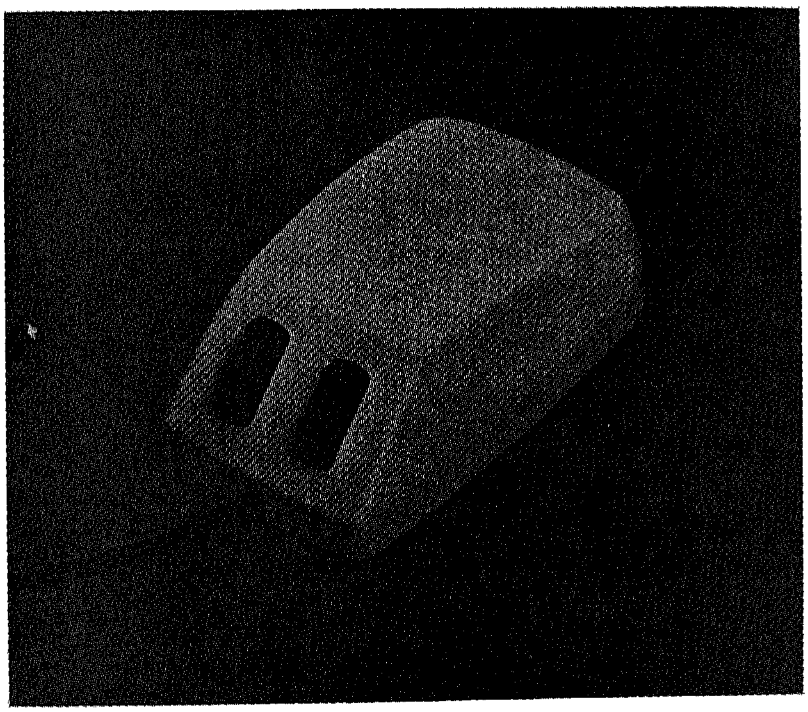
ويعرف السطر 170 برنامجاً فرعيًّا بسيطاً يطبع نجمة (*) فى موضع المؤشر الحالى . وهذا البرنامج الفرعى يتم تنشيطه حينما نضغط على الزر 1 لعصا التوجه أثناء تنفيذ الحلقة التكرارية الرئيسية . وبالمثل فإن السطر 200 يعرف البرنامج الفرعى الذى يطبع مسافة فارغة عند موضع المؤشر الحالى (وبذلك يتم مسح أى شيء آخر موجود عند هذا الموضع) . ويتم تنشيط هذا البرنامج الفرعى بالضغط على الزر 2 أثناء تنفيذ الحلقة التكرارية الرئيسية .

لاحظ أن هذا البرنامج يستمر فى التنفيذ بغير حدود ، أو إلى أن يخرج الحاسب من البيسك بواسطة بعض أوامر من المستخدم . ويمكن تحسين البرنامج بإضافة برنامج فرعى ينهى الحسابات عندما نضغط على مفتاح دالة كما فى مثال ١٠ ــ ١١ .

وبالرغم من أن القلم الضوئى وعصا التوجيه من وحدات الإدخال الشائعة للحاسب الدقيق ، إلا أنه لا يمكن اعتبارهما بأى حال الأجهزة الوحيدة المتاحة ، فالحاسبات الدقيقة الحديثة تدعم أجهزة التحويل الرقمية وأجهزة التعرف على الكلام والعديد من الأنواع الأخرى لأجهزة الإدخال .

ومن الأمور ذات الأهمية الخاصة تزايد الاهتمام باستخدام جهاز يعرف بالمتجول (Mouse) . وهو جهاز لتحديد موضع مشابهاً لعصا التوجيه . وهذا المتجول يسهل التحكم فيه ، وذلك بتحريك المؤشر بواسطة انزلاق المتجول على سطح أملس عريض (عبارة عن مساحة حوالى قدم مربع واحد) بدلاً من عمل تصحيحات دقيقة ليد عصا التوجيه الصغيرة والحساسة .

الشكل ١٠ ــ ١١ يبين صورة فوتوغرافية لمتجول نموذجى . لاحظ أن هناك زرَّيْن عند مقدمة المتجول (وبعض هذه المتجولات لها ثلاثة أزرار ، وأخرى لها زر واحد فقط) . وكما يحدث فى عصا التوجيه فإن المتجول يسمح بتنشيط مختلف الدوال فى أى وقت وذلك ببساطة بالضغط على أحد هذه الأزرار .



شكل ١٠ ــ ١١

وقد أصبح استخدام المتجول شائعاً فى تطبيقات ميكنة بعض المكاتب خصوصاً لمعالجة الكلمات والتطبيقات التى تستخدم بكثرة البيانيات . وتدعم بعض نسخ بيسك الحاسب الدقيق استخدام المتجول (ومع مجموعة من الأوامر تماثل تلك المستخدمة مع عصا التوجيه) والأخرى لا تقوم بذلك . ومع كل فتضمين أوامر المتجول فى بيسك الحاسب الدقيق لابد أن يزداد شيوعاً فى السنوات القادمة .

١٠ ــ ٦ استخدام اللون والصوت USE OF COLOR AND SOUND

تدعم بعض الحاسبات الدقيقة استخدام شاشات عرض TV الملونه ، فتسمح لنص بلون واحد أن يعرض على خلفية بلون مختلف . ويستخدم عرض النص الملون فى إعطاء شكل مؤثر لتحسين البرنامج ، وعلى الأخص إذا كان تفاعل البرنامج عالياً .

ولتلك الحاسبات الدقيقة التى تدعم استخدام اللون يسمح البيسك باختيار الألوان بواسطة جملة COLOR . وهذه الجملة تحتوى عادة على ثلاثة معالم ، تحدد لون الواجهة الأمامى (النص) ، ولون الخلفية ولون الحافة (أى المساحة الخلفية على طوال الحدود الخارجية للشاشة) . وعلى سبيل المثال .. فعلى حاسب IBM الشخصى يمكن اختيار الألوان الخلفية كالآتى

0	black	8	gray
1	blue	9	light blue
2	green	10	light green
3	cyan	11	light cyan
4	red	12	light red
5	magenta	13	light magenta
6	brown	14	yellow
7	white	15	high intensity white

ويمكن اختيار أى لون من الألوان الثمانية الأولى (0 إلى 7) للخلفية أو للحافة . وعلى ذلك فإن جملة COLOR 7,0,0 تدل على نص أبيض على خلفية سوداء وحافة سوداء ، بينما COLOR 0,7,7 تختار نصا أسود على خلفية بيضاء وحافة بيضاء ، Color 14,1,4 تبين نصا أصفر على خلفية زرقاء بحافة حمراء وهكذا . ويمكن تعديل قيمة المعالم فى أى وقت أثناء تنفيذ البرنامج ، ويتسبب هذا فى تغيير الألوان حسب المطلوب .

وعادة ما يسبق استخدام جملة COLOR جملة SCREEN والتى تحدد الأسلوب (النص مقابل البيانيات) وحاله اللون (ممكن أو غير ممكن) . وعلى ذلك فعلى حاسب IBM الشخصى تحدد جملة SCREEN 0,1 أسلوب النص مع إمكان اللون .

مثال ١٠ ــ ١٤ النص المتعدد الألوان Multicolored Text

لتوضيح استخدام نص ملون على حاسب IBM الشخصى نعرض برنامج كتب بميكروسوفت بيسك يعرض على الشاشة نفس السطر من النص مكرراً ، ولكن بمعالم ألوان مختلفة . وينتج عن هذا لف رأسى على الشاشة مع عرض كل سطر بلون مختلف . وبداية فإن .. كل من الخلفية والحافة تكون سوداء . ومع كل .. فإن البرنامج يحتوى على وسيلة لتغيير ألوان الخلفية والحافة فى أى وقت ببساطة وذلك بالضغط على مفتاح الدالة F1 . ويستخدم أيضاً مفتاح الدالة F2 لإيقاف الحسابات .

ويبين شكل ١٠ ــ ١٢ البرنامج كاملاً . ويبدأ بالأمر SCREEN فى السطر 30 الذى يحدد أسلوب النص مع إمكان اللون . والسطر 40 يحذف العرض على الشاشة لأى تعاريف سابقة لمفتاح دالة فى السطر الأسفل للشاشة . وتولد السطور 50 إلى 70 رسالة فى أسفل الشاشة والتى تصف الاستخدام الحالى لمفاتيح الدالة أى :

F1: Change background color F2: End

وتحتاج هذه المجموعة الأخيرة من الأوامر إلى بعض التفسيرات الإضافية . وعلى حاسب IBM الشخصى فإن السطر الأسفل المعروض على الشاشة (أى السطر 25) هو سطر لا يلف ويستخدم عادة لعرض تلقينات ورسائل من هذا النوع . وعادة ما تعرض مثل هذه الرسائل بألوان معكوسة (نص أسود على خلفية بيضاء) وذلك لتمييزهما عن أى شيء آخر قد يظهر على الشاشة . وعلى ذلك يقوم السطر 50 بوضع المؤشر فى أسفل الركن الأيسر من الشاشة ويحدد لوناً عكسيًّا . ويولد السطر 60 الرسالة الحقيقية على السطر الأسفل ، ويحتفظ السطر 70 باللون المعتاد (نص أبيض على خلفية سوداء) ثم يعيد وضع المؤشر أعلى الركن الأيسر للشاشة .

```
10 REM *** COLOR TEXT DEMO ***
20 '
30 SCREEN 0,1
40 KEY OFF
50 LOCATE 25,1 : COLOR 0,7
60 PRINT "F1: Change background color     F2: End ";
70 COLOR 7,0 : LOCATE 1,1
80 ON KEY(1) GOSUB 210
90 ON KEY(2) GOSUB 250
100 KEY(1) ON : KEY(2) ON
110 FOREGROUND=1 : BACKGROUND=0
120 '
130 REM *** MAIN LOOP ***
140 '
150 COLOR FOREGROUND,BACKGROUND,BACKGROUND
160 PRINT "Programming with BASIC is lots of fun! "
170 FOREGROUND=(FOREGROUND+1) MOD 16
180 GOTO 150
190 '
200 REM *** SUBROUTINE TO CHANGE BACKGROUND COLOR ***
210 BACKGROUND=(BACKGROUND+1) MOD 8
220 RETURN
230 '
240 REM *** SUBROUTINE TO TERMINATE THE COMPUTATION ***
250 COLOR 7,0,0
260 END
```

شكل ١٠ ــ ١٢

وتحدد السطور 80 إلى 100 استخدام مفاتيح الدالة كما هو مبين فى مثال ١٠ ــ ١١ ، ثم تتحدد خطة اللون المبدئى (نص أزرق على خلفية سوداء) فى سطر 110 .

ويتم توليد لف النص بحلقة تكرارية تتكون من السطور 180 . وكل مرور خلال الحلقة التكرارية يولد سطراً من النص ، وكل سطر فى النص سيتم عرضه بلون مختلف . وعلى ذلك فالسطر 150 يختار الألوان الحالية حسب القيم المخصصة لمعالم FOREGROUND (للون النص) وBACKGROUND (للخلفية والحافة) . ويولد السطر 160 السطر الحقيقى من النص ، أى

Programming with BASIC is lots of fun!

ويولد السطر 170 قيمة جديدة للمعلمة FOREGROUND للإعداد للمرور التالى خلال الحلقة التكراريه.(لاحظ أن FOREGROUND تأخذ قيماً صحيحة متتابعة بين 0 و15 وذلك لاستخدام المعامل MOD).وفى النهاية يقفل السطر 180 الحلقة التكرارية ، وذلك بالعودة للسطر 150 .

ويحدد السطران 210 و220 البرنامج الفرعى الذى يتم تنشيطه بمفتاح الدالة F1 . ويسبب هذا البرنامج الفرعى زيادة قيمة BACKGROUND بمقدار 1 كلما ضغطنا على F1 .(لاحظ أن BACKGROUND تأخذ قيما صحيحة متتابعة بين 0 و7) . وهذا يسبب بدوره تغيير لون الخلفية والحافة بمجرد عودة التحكم إلى الحلقة التكرارية الرئيسية .

وبالمثل يعرف السطران 250 و 260 البرنامج الفرعى الذى يتم تنشيطه بواسطة مفتاح الدالة F2 . وهذا البرنامج الفرعى يقوم ببساطة بالاحتفاظ بنظام اللون القياس (نص أبيض ، خلفية سوداء ، حافة سوداء) وينهى الحسابات .

وبمجرد البدء فى التنفيذ يبقى البرنامج فى الحلقة التكرارية الأساسية بدون تحديد إلى أن يتم الضغط على مفتاح الدالة F2 ويظهر كل سطر من النص بلون مختلف ، إلا أن الألوان تكرر نفسها كل 16 سطراً حيث يوجد 16 فقط من الألوان الأمامية المختلفة . وسيكون هناك سطر واحد فى كل مجموعة من 16 غير قابل للقراءة ، وذلك لأن اللون الأمامى يكون مماثلاً للألوان الخلفية . ويتغير لون الخلفية كل مرة يتم فيها الضغط على F1 . وتبدأ سلسلة الألوان الخلفية فى التكرار بعد ثمانية ألوان مختلفة حيث أن هناك ثمانية ألوان خلفية متاحة فقط .

وعلى القارىء أن يشغل البرنامج على نوع من حاسب IBM مجهزا بشاشة ملونة إذا أمكن ذلك . وهذا يعطى للقارىء تفهماً أكبر للتأثير الذى ينشأ .

وإذا كان الحاسب مجهزاً بشاشة عرض TV ذات لون واحد (أى أسود وأبيض أو أسود وأخضر) بدلاً من شاشة عرض ملونة فإن جملة COLOR يمكن تفسيرها تفسيراً مختلفاً نوعاً ما . وعلى الأخص فإنه قد يتم استخدام معالم معينة لتحديد خواص النص مثل وضع خط تحته ، كثافة عالية أو نص ومضى وسنرى مثالاً لذلك فى مثال ١٠ ــ ١٧ .

ونعيد انتباهنا الآن لاستخدام الصوت فى برنامج بيسك فمعظم الحاسبات الدقيقة تحتوى على بوق داخلى يمكن تنشيطه تحت تحكم البرنامج . رغم أن هذه الأبواق تكون صغيرة إلا أنها يمكنها إصدار أنواع مختلفة ملحوظة من الأصوات . ويمكن تحسين العديد من برامج البيسك عن طريق استخدام مثل هذه البرامج التى تولد الأصوات .

وعلى سبيل المثال توجد جملة BEEP على حاسب IBM الشخصى والتى تنبه ببساطة البوق لاحداث صوت قصير ذى نغمة عالية بتردد ثابت . وتفيد مثل هذه التنبيهات بلفت انتباه المستخدم لبعض الأحداث المعينة التى يمكن أن تحدث أثناء تنفيذ البرنامج . وعادة ما يستخدم هذا التنبيه فيما يتصل ببرنامج الأخطاء (انظر فصل ١١) .وهناك فوائد أخرى لها أيضا .

مثال ١٠ ــ ١٥

دعنا نعدل برنامج ميكروسوفت بيسك الموضح فى المثال السابق بحيث يقوم البوق بالتنبيه عندما يتغير لون الخلفية (أى عندما يتم الضغط على مفتاح الدالة F1) . وهذا يمكن أن يتم بسهولة جداً ، وذلك بإضافة جملة BEEP للبرنامج الفرعى بالسطور 210 إلى 220 . وعلى ذلك فإن البرنامج الفرعى يمكن كتابته كالآتى

```
200 REM *** SUBROUTINE TO CHANGE BACKGROUND COLOR ***
210 BACKGROUND=(BACKGROUND+1) MOD 8
215 BEEP
220 RETURN
```

(لاحظ إضافة سطر 215) . ولابد للقارىء أن يجرى هذا التغيير وأن ينفذ هذا البرنامج إذا كان ذلك ممكناً بأى حال للتعرف على التأثير الذى ينشأ .

ويحتوى أيضاً ميكروسوفت بيسك على جملة SOUND ، والتى تستخدم لإيجاد صوت فترته وتردده متغيرين . وكلما كان التردد عالياً ، كانت نغمة الصوت عالية . وبالمثل كلما طالت الفترة طال الصوت . وعلى سبيل المثال فإن جملة SOUND 5000,1 تنتج صوتاً قصيراً ذا نغمة عالية نسبياً ، بينما SOUND 100,2000 تولد نغمة طويلة ومنخفضة (تمثل المعلمة الأولى التردد ، والثانية تمثل الفترة) .

ويمكن بتجريب عدد معين من جملة SOUND يتضح ذلك كثيراً حيث أنه يمكن الحصول على عدة أصوات مشوقة باستخدام هذه الجملة . ويكون هذا صحيحاً على الأخص إذا كانت الجملة تدخل ضمن الحلقة التكرارية FOR—TO بقيم متغيرة للمعالم . وبعض هذه التأثيرات موضحة فى المثالين التاليين .

مثال ١٠ ــ ١٦ برمجة البوق Programming the Speaker (A Siren)

يمثل شكل ١٠ ــ ١٣ برنامج ميكروسوفت بيسك مكتوب لحاسب IBM الشخصى . وهو يبين كيف يولد البوق الداخلى صوت السرينة . ويحتوى البرنامج أساساً على الحلقة التكرارية FOR—TO التى تتكرر عددا غير محدود من المرات . وفى كل مرة يتم فيها تنفيذ الحلقة التكرارية ينتج عنها صراخ واحد من السرينة . وهذا الصوت المخصوص فى السرينة يتكرر كل مرة يتم فيها تنفيذ الحلقة التكرارية بالكامل . وتستمر السرينة بغير حدود حتى يتم الضغط على مفتاح الدالة F1 .

دعنا نختبر البرنامج بالتفصيل . فالسطر 30 يمسح عرض أى تعاريف سابقة لمفتاح الدالة من أسفل الشاشة ، ويولد السطران 40 و 50 الرسالة

```
Press F1 to STOP
```

فى مركز الشاشة . والسطر 60 يصاحب مفتاح الدالة F1 ببرنامج فرعى لانهاء الحسابات والسطر 70 ينشط هذه المصاحبة.ويتحدد طول كل صوت منفرد بتخصيص القيمة 0.02 إلى DURATION فى السطر 80.(لاحظ أن كل صوت منفرد يكون قصيراً) .

والسطور 120 إلى 150 تنتج صوتاً حقيقياً للسرينة . وفى السطور 120 إلى 140 نرى الحلقة التكرارية FOR—TO التى تنتج صراخاً واحداً للسرينة وكل مرور خلال هذه الحلقة التكرارية يولد نغمة قصيرة يتحدد ترددها بالدليل (FREQUENCY) . لاحظ أن النغمة سيزداد ترددها أثناء المرور كل مرة خلال الحلقة التكرارية . وحيث أن النغمات تكون قصيرة ، فإن التأثير يكون كما فى حالة إيجاد صوت سرينة يرتفع باستمرار بالرغم من أن الأنغام المفردة لها ترددات غير مستمرة ومتصلة وفى النهاية يجعل السطر 150 الحلقة التكرارية تبدأ من جديد مولدة بذلك تكرار صوت السرينة .

```
10 REM *** SOUND DEMO (SIREN) ***
20 '
30 KEY OFF
40 CLS : LOCATE 12,32
50 PRINT "Press F1 to STOP";
60 ON KEY(1) GOSUB 180
70 KEY(1) ON
80 DURATION=.02
90 '
100 REM *** MAIN LOOP ***
110 '
120 FOR FREQUENCY=400 TO 1000 STEP 5
130     SOUND FREQUENCY,DURATION
140 NEXT FREQUENCY
150 GOTO 120
160 '
170 REM *** SUBROUTINE TO TERMINATE THE COMPUTATION ***
180 END
```

شكل ١٠ ــ ١٣

وتستمر العملية بالكامل حتى يتم الضغط على مفتاح الدالة F1 . وهذا يقوم بتحويل التحكم فى الحال إلى جملة END فى السطر 180 مسبباً إيقاف البرنامج .

ومرة أخرى فإننا نشجع القارىء على تشغيل هذا البرنامج فعلاً حتى يمكنه أن يتفهم ماذا يحدث .

والمثال التالى يوضح الاستخدام المركب لجمل COLOR ، BEEP و SOUND لتحسين برنامج بيسك الذى يولد عرضا بسيطا لنص . وهذا البرنامج يتم تنفيذه على شاشة عرض TV بلون واحد . وعلى ذلك فإن جملة COLOR تستخدم لاختيار خواص مختلفة للنص مثل الكثافة العالية أو الومضية بدلاً من الألوان الحقيقية .

مثال ١٠ ــ ١٧ برمجة شاشة عرض TV Programming a TV Display

شكل ١٠ – ١٤ يبين تغيراً فى البرنامج الأصلى الذى سبق عرضه فى مثال ١٠ – ١٠ الذى يتسبب فى ظهور الرسالة

GO WRONG! GO WRONG! GO WRONG! GO WRONG! GO WRONG! GO WRONG! GO WRONG!

يملأ شاشة من 80 حرف والتى تلف رأسياً . ويكون المنطق هو نفسه أساساً كما فى البرنامج الذى تم عرضه سابقاً (أنظر شكل ١٠ ــ ٣) . ومع كل .. فهذه النسخة من البرنامج يتم تحسينها بإضافات عديدة من جمل COLOR والعديد من جمل BEEP و SOUND فمثلاً .. تسبب جملة COLOR 15,0 فى السطر 40 الرسالة الافتتاحية .

```
WELCOME TO THE WONDERFUL
WORLD OF MICROCOMPUTERS!
```

التى يتم عرضها بكثافة عالية والتى تعطى تأكيداً خاصاً على هذه الرسالة . وبالمثل فإن جملة COLOR 15,0 فى السطر 230 تسبب الكلمة

DARN!

فى الظهور بكثافة عالية معطية ثانياً تأكيداً خاصاً . (لاحظ أن النص ذا الكثافة العالية يختفى بواسطة جملة COLOR 7,0 فى السطرين 90 و280) .

ويمكن الحصول على تأثير مختلف نوعاً ما بجملة COLOR 16,7 فى السطر 150 . وهذه الجملة تسبب عرض السطور التى تحتوى على الرسالة .

GO WRONG! GO WRONG! GO WRONG! GO WRONG! GO WRONG! GO WRONG! GO WRONG!

وذلك كحروف سوداء ومضيه على خلفية بيضاء (أى على شكل عرض ومضى معكوس) . وعلى ذلك يضيف لمسة درامية أثناء لف السطور إلى أعلى الشاشة .

ويمكن تحسين البرنامج باستخدام جملتى BEEP فى السطرين 40 و90 وجمل SOUND فى السطرين 210 و260 . وتسبب جملتا BEEP قيام البوق بالتنبيه عندما يبدأ عرض افتتاحية كل رسالة ملفتاً الانتباه لهذه الرسالة . وبالمثل فإن جملتى SOUND فى السطر 210 تخلقان نغمة منخفضة وتنبيهاً أطول مصاحباً لكل سطر من سطور النص التى تلف (وتقوم أول جملة SOUND بتوليد الصوت الحقيقى) . كما تولد جملة SOUND الثانية وقفة (أى صوت بتردد مرتفع جداً لا يمكن سماعه) . وبدون هذه الوقفة فإن الأصوات التى تنتج بواسطة جملة SOUND الأولى ستستمر معاً أثناء المرور المتتابع فى الحلقة التكرارية FOR—TO محدثة بذلك نغمة واحدة طويلة مستمرة .

```
10   REM  *** TV-DISPLAY DEMO (IBM Version 2) ***
20   '
30   KEY OFF
40   CLS : COLOR 15,0 : LOCATE 11,28 : BEEP
50   PRINT "WELCOME TO THE WONDERFUL"
60   LOCATE 14,28
70   PRINT "WORLD OF MICROCOMPUTERS!"
80   FOR I=1 TO 3000 : NEXT I
90   CLS : COLOR 7,0 : LOCATE 11,18 : BEEP
100  PRINT "Just relax, enjoy yourself and remember that"
110  LOCATE 14,25
120  PRINT "NOTHING CAN POSSIBLY GO WRONG!"
130  FOR I = 1 TO 3000 : NEXT I
140  LOCATE 24
150  COLOR 16,7
160  FOR I = 1 TO 16
170     FOR J = 1 TO 8
180        PRINT "GO WRONG! ";
190     NEXT J
200     PRINT
210     SOUND 500,5 : SOUND 32767,5 : PRINT
220  NEXT I
230  COLOR 15,0
240  CLS : LOCATE 11,37
250  PRINT "DARN!"
260  FOR I=1 TO 18 : SOUND 50,1 : SOUND 32767,1 : NEXT
270  LOCATE 15,30
280  COLOR 7,0 : PRINT "Something went wrong!"
290  LOCATE 23
300  END
```

شكل ١٠ — ١٤

ونرى فى السطر 260 جملتى SOUND متتابعتين كما فى السطر 210 والتى تضمنتها الحلقة التكراريه FOR — TO هو تنفيذ جملتى SOUND هذه مرة بعد الأخرى . ويكون التأثير العام هو عبارة عن سلسلة من الأصوات القصيرة ذات النغمة المنخفضة وتشبه نوعاً ما نقيق الضفدعة . وصوت هذا النقيق يقصد به تحسين كلمة DARN! التى تتولد فى السطر السابق .

ومرة أخرى نشجع القارىء على تنفيذ هذا البرنامج فعلاً حتى يمكنه أن يتفهم بالكامل التأثيرات البصرية والسمعية التى تنتج .

١٠ ـــ ٧ تنقيح البرنامج PROGRAM EDITING

معظم نسخ ميكروسوفت بيسك تسمح بمرونة أكبر فى تنقيح البرنامج أكثر عن النسخ التقليدية من اللغة . فمثلاً فى ميكروسوفت بيسك يمكن أن يتم تعديل برنامج بيسك بثلاث طرق مختلفة باستخدام جملة EDIT مع خاصية تعديل الشاشة الكاملة ومع المنقح التقليدى للنص أو معالج الكلمات .

وتكون جملة EDIT أكثر ملاءمة لإجراء تصحيحات للسطور المفردة فى أجزاء مختلفة من البرنامج . ولاستخدام هذه الجملة يمكن للمبرمج ببساطة كتابة EDIT ويتبعها رقم السطر ، أى

EDIT 50

ويتسبب هذا فى عرض السطر على شاشة عرض TV ويكون المؤشر فى وضعه عند أول السطر . ويمكن إدخال دالة التنقيح المطلوبة (أى حركة المؤشر ادخال ، حذف ... الخ) بالضغط على المفتاح المناسب .

ويمكن للقراء المهتمين الرجوع إلى الكتيب المرجعى للمبرمج فيما يخص حاسباتهم الخاصة للحصول على معلومات أكثر عن هذه الخاصية المفيدة .

مثال ١٠ ـــ ١٨

افترض فى برنامج البيسك الموجود فى مثال ١٠ – ١٧ (شكل ١٠ – ١٤) أن السطر 150 قد تم إدخاله بطريق الخطأ مثل

150 COLLR 16,7

(لاحظ أن COLLR قد أدخلت بدلاً من COLOR) . ولاجراء التعديلات اللازمة يقوم المبرمج بكتابة

EDIT 150

وهذا يتسبب فى ظهور السطر الأصلى (الغير صحيح) على شاشة عرض TV مع ظهور المؤشر تحت 1 فى 150 أى

150 COLLR 16,7

وبعد ذلك يتم تحريك المؤشر إلى الحرف الغير صحيح أى

160 COLLR 16,7

ثم يتم استبدال O بدلاً من L . وبمجرد أن يتم التغيير يقوم المبرمج بالضغط على مفتاح RETURN . وهكذا يستبدل السطر الأصلى بالسطر المنقح .

وإذا كان من الضرورى تغيير عدة سطور متتالية ، فقد يكون من الأنسب طباعة قائمة لكتلة من السطور كاملة على الشاشة بدلاً من تنقيح كل سطر منفرداً ، حيث يمكن القيام بالتغيير المطلوب وذلك بتحريك المؤشر إلى المكان المناسب فى كل سطر ثم نعيد الطباعة حسب الضرورة . ويتسبب الضغط على مفتاح RETURN بعد كل تغيير فى أن يستبدل السطر الأصلى بالسطر المنقح حديثاً . وعلى ذلك فإن المؤشر يأخذ الوضع فى أول السطر التالى أوتوماتيكياً . ويشار إلى هذا بتنقيح الشاشة بالكامل .

مثال ١٠ ـــ ١٩

نعتبر مرة أخرى برنامج البيسك المعروض فى مثال ١٠ ـــ ١٧ . ولنفرض أننا نعرف أن السطور من 160 إلى 220 تحتوى على عدة أخطاء ، فإن إحدى الطرق لتصحيح هذه الأخطاء أن نصحح كل سطر على حدة كما فى المثال السابق . وقد يكون من الأسهل كتابة

LIST 160–220

```
160 FUR I=1 TO 16
170     FOR J=1 TO 16
180     PRUNT "GO WRING! ";
190     NEXT I
200     PRNT
210     SOUND 500 : SOUND 32767 : PRINT
220 NEXT J
```

وينتج عن هذا عرض الآتى على شاشة عرض TV (لاحظ أن كل سطر يحتوى على خطأ واحد على الأقل) ولتصحيح كتلة الجمل هذه نحرك المؤشر أولاً إلى المكان المناسب فى السطر 160 ونصحح الخطأ (FUR تكون FOR) ثم نضغط على مفتاح RETURN . وهذا يحرك المؤشر أتوماتيكياً إلى بداية السطر 170 . نحرك المؤشر ونصحح الخطأ فى السطر 170 ونضغط على مفتاح RETURN ، ثم نصحح الأخطاء فى السطر 180 وهكذا حتى تتم جميع التغييرات . وفى نهاية هذه العملية ، ستظهر السطور من 160 إلى 220 كالآتى :

```
160 FOR I=1 TO 16
170     FOR J=1 TO 8
180     PRINT "GO WRONG! ";
190     NEXT J
200     PRINT
210     SOUND 500,5 : SOUND 32767,5 : PRINT
220 NEXT I
```

لاحظ أنه لم تكن هناك ضرورة لكتابة EDIT 160 ,*EDITé)* ... الخ لكل سطر من السطور المطلوب لتنقيحها ويمكن أيضاً تنقيح برنامج بيسك باستخدام منقح نص تقليدى ، أو معالج الكلمات بشرط أن يكون مخزناً كما فى ملف ASCLL بدلا من أن يكون مكودًا أو مضغوطاً وحتى تحتفظ ببرنامج بهذه الطريقة عندما تستخدم ميكروسوفت بيسك ، فعلى المستخدام أن يطبع .

SAVE "*program name*",A

ويمكن أن يستدعى البرنامج إذاً بواسطة منقح النص ، ويتم التغيير الضرورى .

مثال ١٠ — ٢٠

نفرض أن برنامج "GO WRONG" المعروض فى مثال ١٠ — ١٧ قد تم إدخاله على الحاسب لأول مرة ، ولدينا شك فى وجود عدة أخطاء مطبعية فى كل مكان بالبرنامج . وهذه الأخطاء يمكن تصحيحها باستخدام أى من الطريقتين الموضحتين فى المثالين السابقين . ومع كل فقد يكون من الأسهل حفظ البرنامج كملف ASCII ثم إجراء التصحيحات باستخدام منقح النص .

SAVE "B:DEMO",A

ولتخزين البرنامج كملف ASCII على محرك قرص "B" وإجراء التصحيحات الضرورية . وبمجرد إتمام كل التصحيحات يمكن تخزين الملف الجديد مرة أخرى على المحرك B باسم DEMO. BAS وتنفيذه فيما بعد .

أسئلة للمراجعة
Review Questions

١٠ - ١ حدد نوع نظام التشغيل المتاح على الحاسب الدقيق الموجود فى منزلك أو مكتبك . حدد أيضاً كيفية تمييز الملفات مع نظام التشغيل هذا .

١٠ - ٢ ما هو الفرق بين اسم ملف وامتداد ملف ؟ ما هو الغرض المفيد الذى يؤديه امتداد ملف ؟.

١٠ - ٣ كيف يمكن لبرنامج مخزن على أحد أجهزة التخزين الكبيرة أن يتداول ملف (أى ملف بيانات) والمخزن على جهاز تخزين كبير آخر ؟

١٠ - ٤ أشرح الغرض من كل من أوامر نظام الحاسب الدقيق الآتية :ـــ

AUTO, CLEAR, CONT, DELETE, EDIT, FILES, KILL, LIST, LLIST, LOAD, MERGE, NAME, NEW, RENUM, RUN, SAVE, SYSTEM, TRON, TROFF.

١٠ - ٥ ما هى أوامر النظام المتاحه على الحاسب الدقيق التى تستخدمه فى منزلك أو مكتبك ؟
قارن كل من هذه الأوامر مع تلك الموجوده فى سؤال ١٠ - ٤ .

١٠ - ٦ كم من النصوص (ما عدد الصفوف والأعمده) يمكن عرضها على شاشه عرض TV لحاسب دقيق نموذجى ؟

١٠ - ٧ كيف يمكن مسح الشاشه أثناء تنفيذ برنامج بيسك ؟

١٠ - ٨ كيف يمكن تحريك المؤشر بينما يتم تنقيح تصحيح البرنامج ؟ وكيف يمكن تحريكه أثناء تنفيذ البرنامج ؟

١٠ - ٩ هل يمكن تحريك المؤشر الى موضع على الشاشه المعروض فعلا عليها نص أثناء تنفيذ برنامج بيسك ؟ وإذا تم ذلك فهل هذا يؤدى ذلك إلى اتلاف النص الذى سبق عرضه ؟

١٠ - ١٠ ما هو الفرق بين دوال البيسك SPACE$, SPC, TAB . متى يكون أفضل استخدام لكل داله من هذه الدوال ؟

١٠ - ١١ ما معنى اللف الرأسى ؟ وكيف يمكن التحكم فيه ؟

١٠ - ١٢ لماذا يكون تأخير الوقت مفيداً فى برنامج بيسك للحاسب الدقيق ؟ وكيف يمكن توليد مثل هذا التأخير للوقت ؟

١٠ - ١٣ قارن بين استخدام تأخيرات الوقت المحدوده (أى تأخيرات الوقت التى تدوم لفترة معينه من الوقت) مع عدم تحديد وقت للتأخيرات (الانقطاعات التى يبقى تأثيرها حتى يضغط المستفيد على مفتاح ، الخ) . تحت أى ظرف يكون كل نوع من تأخير الوقت أكثر مناسبه ؟ .

١٠ - ١٤ أشرح المجموعات الرئيسية للمفاتيح على لوحة مفاتيح لحاسب دقيق معين . ما الغرض من كل مجموعة رئيسية ؟

١٠ - ١٥ هل تتضمن نسخه البيسك المتاحه على الحاسب الدقيق الخاص بك امكانيه برمجه مفاتيح الداله ؟ وإذا كان كذلك فكيف يمكن أن يتم هذا ؟

١٠ - ١٦ ما هو القلم الضوئى ؟ ولأى نوع من التطبيقات تكون هذه الوسيله أكثر فائدة ؟

١٠ - ١٧ هل تتضمن نسخه البيسك المتاحه على الحاسب الدقيق الخاص بك امكانيه برمجه قلم ضوئى ؟ وإذا كانت كذلك فكيف يمكن أن يتم هذا ؟

١٠ - ١٨ ما هى عصا التوجيه ؟ ولأى نوع من التطبيقات تكون هذه الوسيله أكثر فائدة ؟

١٠ - ١٩ هل تتضمن نسخه البيسك المتاحه على الحاسب الدقيق الخاص بك امكانية برمجة عصا توجيه ؟ وإذا كان كذلك فكيف يمكن أن يتم هذا ؟

١٠ - ٢٠ ما هو المتجول ؟ ولأى نوع من التطبيقات تكون هذه الوسيله أكثر فائدة ؟ (قارن مع عصا التوجيه) .

١٠ - ٢١ هل تتضمن نسخه البيسك المتاحه على الحاسب الدقيق الخاص بك امكانيه برمجه المتجول ؟ وإذا كان كذلك فكيف يمكن ان يتم هذا ؟

١٠ - ٢٢ ما هو الغرض من جمله COLOR في وسيله النصوص ؟ (بالمقارنه بوسيلة البيانيات) ؟ ولأى حاله تستخدم كل معلمه ؟

١٠ - ٢٣ كيف تفسر جمله COLOR بواسطة الحاسب الدقيق المجهز بشاشه عرض TV ذات اللون الواحد ؟ .

١٠ - ٢٤ ما هو الغرض من جملة SCREEN ؟ وفيم تستخدم كل معلمه عندما تكون في وسيلة النصوص ؟

١٠ - ٢٥ هل يحتوى الحاسب الدقيق الخاص بك على جملتى SCREEN, COLOR ؟ وإذا لم يكن كذلك فهل توجد جمل أخرى تؤدى نفس الشيء ؟

١٠ - ٢٦ ما الغرض من جملة BEEP ؟

١٠ - ٢٧ ما الغرض من جملة SOUND ؟ وما هو الغرض من كل معلمه ؟

١٠ - ٢٨ كيف يمكن استخدام جمل SOUND, BEER, COLOR لتحسين برنامج بيسك ذى طبيعه غير بيانيه ؟

١٠ - ٢٩ أشرح استخدام أمر EDIT . لأى نوع من حالات التنقيح يكون هذا الأمر أكثر مناسبة ؟

١٠ - ٣٠ أشرح كيف يمكن أن يتم التنقيح بإستخدام خاصية تنقيح الشاشه بالكامل الموجوده في كثير من نسخ بيسك الحاسب الدقيق . تحت أى ظروف يكون تنقيح الشاشه بالكامل وفضلا عن تكرار استخدام امر EDIT ؟

١٠ - ٣١ هل نسخه البيسك للحاسب الدقيق الخاص بك تحتوى على أمر EDIT (أو امر مقارن) ؟ وهل يدعم ذلك تنقيح الشاشه بالكامل ؟

١٠ - ٣٢ قارن استخدام منقح النصوص مع استخدام خاصية تنقيح الشاشه بالكامل والمتاحه في كثير من نسخ بيسك الحاسب الدقيق . أيهما أفضل وتحت أى ظروف ؟

١٠ - ٣٣ كيف يمكن حفظ برنامج بيسك على أنه ملف ASCII على نسختك الخاصه من بيسك الحاسب الدقيق ؟

أسئله تكميليه

Supplementary Problems

المسائل الاتيه تهتم بجمع المعلومات بدلاً من الحلول الحقيقيه للمسائل . أجب على الاسئله كما تطبقها على نسخة البيسك للحاسب الدقيق الخاص بك .

١٠ - ٣٤ كيف يمكن تمييز ملف كامل مكتوب للحاسب الدقيق الخاص بك ؟ وكيف يمكن تمييز محركات اقراص متعدده من واحد الى آخر ؟

١٠ - ٣٥ ما هى أوامر النظام المتاحه في نسختك من البيسك ؟ كيف تختلف أوامر النظام هذه عن أوامر النظم التقليدية المشروحه في فصل ٣ ؟ .

١٠ - ٣٦ ما هو اقصى عدد من السطور يمكن عرضه على شاشه عرض TV الخاصه بك وما هو أقصى عدد للحروف لكل سطر ؟

١٠ - ٣٧ كيف يمكن مسح الشاشه من خلال نسختك من البيسك ؟

١٠ - ٣٨ كيف يمكن تحريك المؤشر أثناء تنقيح البرنامج ؟ كيف يمكن تحريك المؤشر من خلال نسختك من البيسك ؟

١٠ - ٣٩ كيف يمكن أن يبدأ اللف الرأسى ويتم التحكم فيه من خلال نسختك من البيسك ؟

١٠ - ٤٠ كيف يمكن توليد ايقاف الوقت مؤقتا (أى الايقاف لفترة وقت محدوده) أثناء تنفيذ برنامج مستخدماً نسختك من البيسك ؟ كيف يمكن توليد ايقاف غير محدد الوقت (أى ايقاف لفترة وقت غير محدده) ؟

١٠ - ٤١ كيف يمكن تحويل المخرجات من شاشه عرض TV لالة طباعة ؟ هل يتضمن هذا مخرجات مصاغه ؟

١٠ - ٤٢ هل تحتوى لوحه مفاتيح حاسبك الدقيق على مفاتيح حركه المؤشر و / أو مفاتيح الداله ؟ إذا كان كذلك هل توجد جمل خاصه تسمح ببرمجة هذه المفاتيح ؟

١٠ - ٤٣ هل حاسبك الدقيق يمدك باستخدام وسائل مساعده خاصه مثل القلم الضوئى أو عصا التوجيه أو المتجول ؟ إذا كان كذلك هل هناك جمل بيسك خاصه تسمح ببرمجة هذه الوسائل ؟

١٠ - ٤٤ هل حاسبك الدقيق يمدك بالنصوص الملونه ؟ إذا كان كذلك كيف يحدد اللون فى البيسك ؟ هل يمكن تحديد لون خلفيه منفصل ؟ هل هناك لون حافه منفصل ؟

١٠ - ٤٥ هل هناك أوامر خاصه لتنبيه البوق فى نسختك من البيسك ؟

١٠ - ٤٦ هل نسختك من البيسك تسمح بتوليد اصوات مختلفه تحت تحكم البرنامج ؟ إذا كانت كذلك ما هى الأوامر التى تستخدم لتنفيذ ذلك ؟ كيف يمكن تحديد التردد والفتره ؟ .

١٠ - ٤٧ هل نسختك من البيسك تحوى امكانية تنقيح سطر ؟ إذا كانت كذلك فكيف يمكن تحريك المؤشر على السطر الذى يتم تنقيحه ؟ كيف يمكن ادخال الحروف وحذفها ؟

اسئله للبرمجه

Programming Problems

معظم المسائل المعطاه فيما يلى تحتاج الى استخدام اللون «Color» إذا لم يكن عندك شاشه ملونه يمكن تبديل الألوان المختلفه بالخواص المختلفه لأحاديه اللون (مثل شدة ضوء عاليه أو عكس أو نص ومضى) لألوان مختلفه .

١٠ - ٤٨ عدل البرنامج المعطى فى مثال ١٠ - ١٤ ليحتوى على استخدام مفاتيح الدوال F1 حتى F4 .

استخدام مفاتيح الدوال بالطرق الآتيه :ـــ

F1 يختار لون الواجهه

F2 يختار لون الخلفيه

F3 يختار لون الحافه

F4 ينهى الحسابات

اجعل الحاسب ينبه «Beep» عندما يتغير اللون ربما تريد أن تبدل رساله برساله من اختيارك .

١٠ - ٤٩ عدل برنامج التحكم فى المخزون (مثال ٩ - ٣١) بحيث يظهر المخزون صفرا بلون مختلف (أى أحمر) عن النصوص الأخرى المعروضه على الشاشه .

١٠ - ٥٠ عدل برنامج توليد اعداد فيبوناشي والبحث عن الأرقام الأوليه (مثال ٩ - ١١) بحيث تظهر الأرقام الأوليه بلون مختلف عن باقى النص . أجعل الحاسب ينبه «Beep» عندما يتم طباعة عدد أولى .

١٠ - ٥١ السؤال ٩ - ٦٣ يطلب منك حل المسألة ٥ - ٥٥ (حساب درجات امتحانات طلبه ، متضمنا متوسط الفصل والانحراف المتوسط لكل طالب عن متوسط الفصل) باستخدام الخصائص المحسنه المتاحه فى نسختك من بيسك الحاسب الدقيق . حل المسأله ٩ - ٦٣ بإضافة استخدام اللون للبيانات المخرجه المعروضه . وأعرض على الأخص كل النص بنفس اللون ما عدا متوسط الفصل والانحراف المتوسط لكل طالب عن متوسط الفصل . واعرض متوسط الفصل بلون آخر والانحراف بلون ثالث .

١٠ - ٥٢ المسأله ٩ - ٥٥ تطلب منك إعادة كتابة مولد بجلاتين والموجود بمثال ٩ - ٢٨ بحيث يمكنها أن تتضمن عدة خصائص اضافيه مثل نص متعدد السطور ، علامات التنصيص وأصوات ثنائية الحروف .

أعد كتابة مولد بجلاتين كما هو مبين فى مسأله ٩ - ٥٥ ، بالاضافه الى ذلك أضف امكانية عرض النصوص الانجليزيه (أى البيانات المدخله) على شاشه عرض TV بلون واحد وعرض بجلاتين (المخرجات المناظره) بلون آخر . استخدم ثلاث مفاتيح دوال مختلفه لتغيير الوان النص الانجليزى ، ونص بجلاتين والخلفيه على الترتيب .

أجعل الحاسب ينبه «Beep» فى بداية كل ترجمة بجلاتين (كل كتلة مخرجات جديده) استخدم مفتاح داله لتحويل التنبيه لتشغيلها أو ابطالها .

١٠ - ٥٣ أعد كتابة البرنامج الخاص بحساب الاستهلاك (مثال ٤ - ٩) بحيث يظهر كل عمود من الاعداد على شاشه عرض TV بلون مختلف . استخدم الخواص المتقدمة التى تعطيها نسختك من بيسك الحاسب الدقيق حيثما أمكن ذلك عمليا . واستخدم على الأخص مفاتيح الداله لاختيار الطريقه التى تستخدم لحساب الاستهلاك (أى استخدم F1 لاختيار طريقه الخط المستقيم للاستهلاك و F2 لاختيار طريقة توازن الانحراف المزدوج للاستهلاك ، F3 لاختيار طريقة مجموع أرقام السنوات و F4 لانهاء الحسابات) . اعرض قائمه مختصره عند أعلى الشاشه لتبين الغرض فى كل مفتاح داله . بين نتائج كل من الحسابات على شاشه منفصله (أى امسح الشاشه بين الحسابات) .

١٠ - ٥٤ عدل برنامج «Hello» المبين فى مثال ١٠ - ١٢ . (برمجه قلم ضوئى) ليحتوى على استخدام لون . أعرض على الأخص قائمه اللغات بلون واحد والتحيه المناظره بآخر . اجعل الحاسب ينبه «Beep» عند عرض التحيه .

١٠ - ٥٥ أعد كتابه البرنامج المعطى فى مثال ١٠ - ١٢ بحيث يمكن اختيار لغه باستخدام مفاتيح الداله بدلا من قلم ضوئى . اجعل البرنامج يحتوى على استخدام لون كما هو موجود فى مثال ١٠ - ٥٤ .

١٠ - ٥٦ أعد كتابه البرنامج المعطى فى مثال ١٠ - ١٢ بحيث يمكن اختيار لغه مع عصا توجيه بدلا من قلم ضوئى . اجعل البرنامج متضمنا استخدام لون كما هو مبين فى مثال ١٠ - ٥٤ .

١٠ - ٥٧ اعد كتابه البرنامج المعطى فى مثال ١٠ - ١٢ بحيث يمكن اختيار لغه مع المتجول بدلا من قلم ضوئى . أجعل البرنامج يحتوى على استخدام لون كما هو مبين فى مثال ١٠ - ٥٤ .

١٠ - ٥٨ حل المسألة ٥ - ٥٦ (اختيار العواصم المقابله لدولها) بإستخدام مفاتيح الداله لاختيار أما الدوله أو العاصمه . تأكد من استخدام الخصائص المحسنه الموجوده فى نسختك من بيسك الحاسبات الدقيق عندمل يمكن ذلك عمليا . استخدم ايضا اللون والصوت كما هو موضح فى مثال ١٠ - ٥٤ .

١٠ - ٥٩ كرر المسألة ١٠ - ٥٨ مستخدما كل من أجهزة الأدخال المبرمجه الآتيه لاختيار أما دوله أو عاصمه .

(أ) قلم ضوئى (ب) عصا توجيه (جـ) متجول

تأكد من تحسين برنامجك بالاستخدام الفعال للصوت واللون .

١٠ - ٦٠ عدل البرنامج الموجود فى مثال ١٠ - ١٣ (برمجه عصا التوجيه) بحيث يمكن استخدام مفاتيح الداله لاختيار الوان مختلفه للواجه والخلفيه . صمم البرنامج بطريقه بحيث يمكن عرض الوان النجوم المختلفه على خلفيه منتظمه . اسمح بامكان تغير لون الخلفيه فى أى وقت .

١٠ - ٦١ أعد مسأله ١٠ - ٦٠ باستخدام متجول بدلا من عصا توجيه .

١٠ - ٦٢ عدل البرنامج المعطى فى مثال ١٠ - ١٦ بحيث يتم توليد صوت سرينه « عالى ومنخفض » . اجعله يحتوى على ما يمكنه تغير الوان الواجه والخلفيه اتوماتيكيا كل مره يتولد فيها صوت السرينه .

١٠ - ٦٣ اكتب برنامج بيسك والذى يولد مقياس موسيقى فى مفتاح C . ضمن البرنامج ما يمكنه اختيار مقياس تصاعدى أو مقياس تناولى أو كليهما (أى المقياس الذى يصعد ثم يهبط) .

اسمح للاختيار بأن يتم بواسطه اختيار مفتاح داله مناسب . أعرض قائمه فى أعلى الجزء الأيسر من الشاشه لتبين الغرض من كل مفتاح داله متاح (ويمكن أن يفيدك كتيب مرجع مخطط البرامج عن الترددات اللازم تحديدها حتى يمكن توليد النغمات الموسيقيه .

١٠ - ٦٤ أكتب برنامج بيسك يمكنه أن يلعب اعنيه بسيطه على بوق حاسبك (ويمكن لكتيب مرجع مخطط البرامج أن ان يبين كيف يمكن توليد نغمات موسيقيه بلغه بيسك لحاسبك المعين) .

الفصـل ١١

البرمجة سهلة الاستخدام .
User-Friendly Programming

تشجع بيئة الحاسب الدقيق الأعلى تفاعلاً على استخدام حوار المستخـدم فى كثير من برامج البيسك . وعادة ما تحتوى هذه الحوارات على صورة من صور تفاعل السؤال الجواب حيث يسأل الحاسب الأسئلة ويقوم المستخدم بتقديم الأجوبة . وتعمل هذه الطريقة بشكل حسن جدا عندما تكون الأسئلة بسيطة وقليلة العدد ، إلا أنه فى بعض التطبيقات قد تكون الأسئلة عديدة أو معقدة أو مكررة مما ينتج عنه تشويش أو ارباك أو إحباط من جانب المستخدم . وفى ظل هذه الظروف يكثر احتمال حدوث أخطاء فى إدخال البيانات . هذا بالإضافة إلى أن المستخدم قد يستخلص أن البرنامج صعب أو مثير للأعصاب بدرجة لا يستطيع معها استخدامه بسهولة وعلى ذلك فقد يتجنب استخدامه فى المستقبل .

ويمكننا غالبا تجنب مثل هذه المشاكل بإدخال خصائص الاستخدامات السهلة داخل البرنامج مثل جمل التلقين والقوائم واختبارات الخطأ وتحقق المستخدم . وتسهل هذه الخصائص عامة استخدام البرنامج مع أن تبنيها عادة مايحتاج إلى جهد كبير من جانب المبرمج .

سنناقش استخدام العديد من خصائص الاستخدامات السهلة فى هذا الفصل .

١١ ــ ١ التلقينات PROMPTS

التلقين هو أمـر مختصـر أو شرح يولده الحاسب قبل طلب المعلومة . وعلى ذلك قد يشرح التلقين نوع المعلومة المطلوبة أو الإجابة المسموح بها . ويتم خلق الحوار بين المستخدم والحاسب من خلال الاستخدام الذكى لمثل هذه التلقينات . وقد قابلنا بالفعل استخدام التلقينات

وقد قابلنا بالفعل استخدام التلقينات فى كثير من الأمثلة الموجودة فى الفصول السابقة من هذا الكتاب . ومع كل فإن معظمها كان مختصراً جداً وسنعيد الآن استخدام تلقينات أكثر تفصيلاً مقترنة ببرامج تصحيح الأخطاء التى تختبر صحة البيانات المدخلة .

مثال ١١ ــ ١

نفرض أنه مطلوب من برنامج البيسك للحاسب الدقيق تلقين المستخدم درجات اختبارات الطلبة والتى تتراوح قيمتها بين 0 و 100 فى المائة . وقد يحتوى البرنامج على جملة INPUT التالية والتى تحتوى على تلقين مقبول .

```
100 INPUT "Exam score (0-100): ",SCORE
```

ويخبر التلقين المستخدم عن المعلومات المطلوبة وتبين مدى القيم المسموح بها.ومع كل .. فالبرنامج لن يقوم باختبار تحديد ما إذا كانت القيمة التى أدخلها المستخدم فعلاً (أى القيمة التى خصصت إلى SCORE) تقع دال المدى المطلوب .

وهنا تلقين مشابه مصحوب باختبار الخطأ للقيمة العددية الغير صحيحة المدخلة .

```
100 LOCATE 3,1: INPUT "Exam score (0-100): ",SCORE
110 IF SCORE < 0 OR SCORE > 100 THEN BEEP: LOCATE 3,1:
    PRINT SPACE$(30): GOTO 100
```

إذا خصص الآن لـ SCORE قيمة عددية خارج المدى المسموح به سوف يبين الحاسب الخطأ وتحذف القيمة السابقة من على الشاشة (فى الحقيقة السطر كله سوف يستبدل بـ 30 مسافة فارغة) وسوف يظهر التلقين ثانياً فى مكانه الأصلى . ويستمر البرنامج فى الدوران فى هذه الحلقة التكرارية إلى أن يدخل المستخدم قيمة مقبولة SCORE .

الجملتان المبينتان سابقاً مناسبتان لتصيد أى قيم عددية غير صحيحة ، إلا أنه تتبقى إمكانية قيام المستخدم بإدخال حرف آخر غير عددى بدلا من عدد . وهنا يكون التلقين مصحوباً باختبار الخطأ الأكثر شمولاً والذى يحتوى على اختبار للحروف غير المرغوب فيها كما يحتوى على اختبار للأرقام غير المناسبة .

```
100 LOCATE 3,1: INPUT "Exam score (0-100): ",ANS$
110 IF LEFT$(ANS$,1)="0" THEN SCORE=0: GOTO 130
    ELSE SCORE=VAL(ANS$)
120 IF SCORE <= 0 OR SCORE > 100 THEN BEEP: LOCATE 3,1:
    PRINT SPACE$(30): GOTO 100
130 . . . next statement . . .
```

الآن تكون استجابة المستخدم هى الإدخال المبدئى كسلسلة حروف (ANS$) بدلاً من قيمة عددية. وإذا ما قام المستخدم بإدخال القيمة 0 فإن البرنامج سوف يتعرف عليه فى السطر 110 ويخصص القيمة 0 لـ (SCORE) ثم ينتقل خارج الحلقة التكرارية إلى السطر 130 . ومع كل فإذا ما أدخل أى شىء آخر فإن قيمة (ANS$) تتحول مباشرة إلى قيمة عددية وتخصص لـ (SCORE) . وسوف تتحول أى قيمة غير عددية أتوماتيكيا إلى 0 ، وتعتبر قيمة غير مقبولة بسبب الاختبار الموجود فى السطر 120 . وعلى ذلك فإنه يمكننا اكتشاف الحروف غير الرقمية والغير مرغوب فيها وكذلك القيم العددية التى لاتقع خلال المدى المسموح به .

مثال ١١ ــ ٢ إدخال درجات امتحان طلبة Entering Student Examination Scores

فى مثال ٩ ــ ٢٩ قابلنا برنامج البيسك الذى ينشىء ملف بيانات متتابعا ، ويحتوى على مجموعة من درجات امتحان الطلبة . والبرنامج الأصلى مكتوب بلغة ميكروسوفت بيسك لحاسب IBM الشخصى المبين فى شكل ٩ ــ ٨ .

```
10 '****** CREATE A SEQUENTIAL DATA FILE ******
20 '
30 '****** (ENTER STUDENT EXAMINATION SCORES - VERSION 2) ******
40 '
50 DIM SCORE(12)
60 OPEN "O",1,"SCORES"
70 KEY OFF: CLS: WIDTH 80
80 LOCATE 1,1: INPUT "Course title: ",TITLE$
90 LOCATE 3,1: INPUT "Term: ",TERM$
100 LOCATE 5,1: INPUT "How many exam scores per student? (1-12) ",ANS$
110 N=VAL(ANS$)
120 IF N < 1 OR N > 12 THEN BEEP: LOCATE 5,1: PRINT SPACE$(60): GOTO 100
130 PRINT #1,TITLE$
140 PRINT #1,TERM$
150 COUNT=1
160 '
170 '*** BEGIN LOOP ***
180 '
190 CLS: PRINT "Student number";COUNT
200 LOCATE 3,1: INPUT "Student name (Type END to end data entry): " ,N$
210 IF N$="END" OR N$="end" THEN 330
220 PRINT #1,N$
230 FOR I=1 TO N
240    LOCATE I+4,1: PRINT "Exam number";I;" (0-100): ";: INPUT "",ANS$
250    IF LEFT$(ANS$,1)="0" THEN SCORE(I)=0: GOTO 270 ELSE SCORE(I)=VAL(ANS$)
260    IF SCORE(I) <= 0 OR SCORE(I) > 100 THEN BEEP: LOCATE I+4,1:
       PRINT SPACE$(60): GOTO 240 ELSE PRINT #1,SCORE(I);
270 NEXT I
280 PRINT #1,""
290 COUNT=COUNT+1: GOTO 190
300 '
310 '*** END LOOP ***
320 '
330 CLOSE
340 END
```

شكل ١١ ــ ١

دعنا الآن نغير هذا البرنامج لنجعله أكثر تحاورا وعلى ذلك يكون أكثر سهولة فى الاستخدام وذلك بإضافة بعض التلقينات الإضافية مع اختبارات الأخطاء المناظرة . ويظهر البرنامج الجديد فى شكل ١١ ــ ١ . ومنطق البرنامج الكلى يشابه ذلك الذى استخدم فى البرنامج السابق . ومع كل.. فقد أضفنا التلقينات واختبارات الخطأ المناظرة لكل من الكميات العددية المدخلة.ونلاحظ على الأخص أن السطور من 100 إلى 120 هى تلقينات لعدد درجات الامتحان لكل طالب وعندئذ نختبر القيم التى أدخلت . وبالمثل فإن السطور من 240 إلى 260 تلقن كل درجة امتحان ثم تختبر القيم التى أدخلت . وتشابه طرق اختبارات إلى حد كبير النموذج المعروض فى مثال ١١ ــ ١ .

وبالإضافة إلى ذلك فإن هذا البرنامج يستخدم عدداً من خصائص التحكم فى الشاشة وهو أكثر عمومية ، بمعنى أنه يمكن أن يقوم بتسكين من 1—12 امتحان لكل طالب . لاحظ استخدام الحلقة التكرارية FOR—TO (السطور من 230 إلى 270) لإدخال كل درجات امتحان الطالب من لوحة المفاتيح وكتابتها فى ملف البيانات المتتابع .

شكلا ١١ ــ ٢ ، ١١ ــ ٣ يوضحان حواراً نموذجياً لمستخدم يحدث عند تنفيذ البرنامج . ويبين شكل ١١ ــ ٢ الحوار المبدئى الذى يلقن عنوان المقرر والفصل الدراسى وعدد الامتحانات لكل طالب . استجابات المستخدم تحتها خط .

```
Course title: Comp Sci 141

Term: Fall 1985

How many exam scores per student? (1-12) 5
```

شكل ١١ ــ ٢

```
Student number 1

Student name (Type END to end data entry): Adams B F

Exam number 1   (0-100): 45
Exam number 2   (0-100): 80
Exam number 3   (0-100): 80
Exam number 4   (0-100): 95
Exam number 5   (0-100): 55
```

شكل ١١ ــ ٣

يبين شكل ١١ ــ ٣ الحوار الذى يطلب اسم ودرجات الامتحان لأول طالب . وهنا أيضاً تكون استجابات المتسخدم تحتها خط . ونفس هذا الحوار سوف يتكرر لكل طالب تال حتى ندخل كلمة END فى مكان اسم الطالب .

ومن سوء الحظ أن هذه الأشكال لايمكنها توضيح آثار اختبارات الخطأ . وعلى ذلك فإن القارىء مطالب بتشغيل هذا البرنامج فعلا وإدخال معلومات غير صحيحة حتى يحصل على بعض الإدراك من حيث طريقة عمل اختبارات الخطأ .

١١ ــ ٢ القوائــم MENUS

تحتاج بعض البرامج التى يختارها المستخدم إلى أحد الاختيارات العديدة والتى قد تكون متاحة . وعادة مايكون أفضل طريقة للاختيار هى الاختيار من القائمة ، أى عرض قائمة بجميع الاختيارات على الشاشة مع رقم أو حرف متميز مناظر لكل اختيار.ويمكن للمستخدم أن يقوم بالاختيار بطريقة مبسطة وذلك بكتابة الحرف أو الرقم المناسب . وعملية الاختيار من القائمة هذه سهلة جداً وتؤدى إلى تقليل احتمال حدوث الأخطاء .

مثال ١١ ــ ٣ .

نفترض أن برنامج ميكروسوفت بيسك المكتوب لحاسب IBM الشخصى يحتوى على إمكان اختيار واحد من أربعة ألوان بخلفيات مختلفة . قد يحتوى البرنامج على الجمل الآتية والتى تولد قائمة وتسمح للمستخدم أن يختار منها .

```
100 CLS
110 LOCATE 1,1: PRINT "Background Colors:"
120 LOCATE 3,1: PRINT "  1 - Blue"
130 LOCATE 4,1: PRINT "  2 - Green"
140 LOCATE 5,1: PRINT "  3 - Red"
150 LOCATE 6,1: PRINT "  4 - White"
160 LOCATE 8,1: INPUT "Please enter your selection: ",ANS$
170 CHOICE=VAL(ANS$)
180 IF CHOICE < 1 OR CHOICE > 4 THEN BEEP: LOCATE 8,1:
    PRINT SPACE$(60): GOTO 160
190 ON CHOICE GOTO 200,300,400,500
```

وعند تنفيذ البرنامج فإن القائمة الآتية سوف تظهر على الشاشة .

```
Background Colors:

  1 - Blue
  2 - Green
  3 - Red
  4 - White

Please enter your selection:
```

سوف يستجيب المستخدم بإدخال 1 أو 2 أو 3 أو 4 . وبناء على ذلك يقوم البرنامج باختيار لون الخلفية الصحيح وذلك بالتفريع للسطور 200 أو 300 أو 400 أو 500 كما توجهه جملة ON—GOTO فى سطر 190 .

لاحظ أن السطور من 160 إلى 180 تحتوى تلقيناً واختباراً للخطأ المناظر الذى يتسبب عندما يتم إدخال أية قيمة غير 1 أو 2 أو 3 أو 4 (بما فى ذلك الحروف غير الرقمية) فى أن ترفض . وعلى ذلك فإنه يسمح للمستخدم بأن يستجيب فقط بإدخال واحد من الاختيارات المعطاة فى القائمة .

وفى هذا المثال بالذات كان يمكننا استخدام الحروف بدلاً من الأرقام للاختيار من القائمة.ويكون الاختيار المنطقى هو استخدام الحرف الأول من كل لون.وعلى ذلك فإن القائمة قد تظهر كما يأتى :—

```
Background Colors:

  B - Blue
  G - Green
  R - Red
  W - White

Please enter your selection: _
```

ويكون هذا مناسباً لهذه المجموعة الخاصة من الألوان ، حيث أن كل لون.يبدأ بحرف مختلف . أما إذا كان هناك لونان أو أكثر يبدآن بنفس الحرف (على سبيل المثال إذا كان black أيضاً متاحاً) فإن استخدام الحروف الأولى فى بنود القائمة لن يكون ممكناً . ومن الممكن اختيار حروف أخرى بحيث يكون كل بند فى القائمة له حرف وحيد . وقد يخلق هذا تشويشاً للمستخدم ولذلك فهو غير مرغوب فيه .

وقد تحتوى بعض البرامج على كل من القائمة الرئيسية وواحدة أو أكثر من القوائم الفرعية . وفى هذه الحالات فقد يكون من المرغوب فيه استخدام كل من الحروف والأرقام كاختيارات من القائمة . وعلى سبيل المثال قد تستخدم الأرقام فى القائمة الرئيسية وتستخدم الحروف فى القوائم الفرعية. ويبين المثال التالى هذه الطريقة .

مثال ١١ ـــ ٤ التمويل الشخصى (حسابات الربح المركب) Personal Finance (Compound Interest Calculations)

تهتم كثير من المسائل فى اقتصاديات المستهلك بحسابات الربح المركب . وعلى الأخص فقد نرغب أحياناً معرفة المبالغ التى سوف تتراكم فى حساب توفير أو كم يتكلف سداد قرض بعد فترة زمنية معينة بمعدل فائدة محدد وعدد فترات مضاعفة الربح .

سوف نعتبر فى هذا المثال ثلاثة حسابات من هذا النوع وهى :-

١ ـــ القيمة المستقبلية لمبلغ محدد .
٢ ـــ القيمة المستقبلية لإيداعات شهرية ثابتة .
٣ ـــ الأقساط الشهرية لسداد القرض .

سوف نقوم ببناء برنامج لقائمة مفردة يسمح بإنجاز كل هذه الحسابات على حاسب IBM الشخصى وسيكتب البرنامج بلغة ميكروسوفت بيسك .

وفيما يلى المعادلات الضرورية لكل نوع من الحسابات . وهذه المعادلات تستخدم الرموز التالية :

P = القيمة الحالية للمبلغ سواء أكان قرض أم وديعة .
F = جملة المبلغ فى المستقبل .
A = الإيداع الشهرى المنتظم أو السداد الشهرى المنتظم .
i = معدل الفائدة السنوى (معبراً عنه كرقم عشرى) .
m = عدد فترات الفائدة لكل سنة .
n = عدد السنوات .

والمعادلات الحقيقية هى :

١ ـــ القيمة المستقبلية (F) لمبلغ محدد (P)

(أ) مضاعفات :

سنوية ، نصف سنوية ، ربع سنوية ، شهرية ، أو يومية (m = 1 أو 2 أو 4 أو 12 أو 365 على الترتيب) .

$$F = P(1 + i/m)^{mn}$$

(ب) مضاعفات مستمرة .

$$F = Pe^{in}$$

٢ ـــ القيمة المستقبلية (F) لسلسلة الإيداعات الشهرية الثابتة (A)

(أ) مضاعفات :-

سنوية ، نصف سنوية ، ربع سنوية ، أو شهرية (m = 1 أو 2 أو 4 و12 على الترتيب) .

$$F = \left(\frac{12A}{m}\right)\left[\frac{(1 + i/m)^{mn} - 1}{i/m}\right]$$

(ب) مضاعفات يومية . (m = 365)

$$F = A\left[\frac{(1 + i/m)^{mn} - 1}{(1 + i/m)^{m/12} - 1}\right]$$

(جـ) مضاعفات مستمرة

$$F = A\left[\frac{e^{in}-1}{e^{i/12}-1}\right]$$

٣ ــ سداد شهرى (A) لقرض (P)

(أ) مضاعفات سنوية او نصف سنوية أو ربع سنوية أو شهرية (m = 1 أو 2 أو 4 أو 12 على الترتيب) .

$$A = \left(\frac{mP}{12}\right)\left[\frac{(i/m)(1+i/m)^{mn}}{(1+i/m)^{mn}-1}\right]$$

(ب) مضاعفات يومية

$$A = P(1+i/m)\left[\frac{(1+i/m)^{m/12}-1}{(1+i/m)^{mn}-1}\right]$$

(جـ) مضاعفات مستمرة

$$A = Pe^{in}\left[\frac{e^{i/12}-1}{e^{in}-1}\right]$$

تصميم البرنامج Design of the Program

سنقوم بتصميم البرنامج التحاورى فى طبيعته والعام حيث أمكن فى الحدود العملية . وسوف يبدأ البرنامج بعرض قائمة رئيسية تسمح للمستخدم باختيار واحد من الأنواع الثلاثة المختلفة للحسابات أو لإنهاء الحساب . عندئذ سوف تخلو الشاشة والبرنامج سوف يلقن البيانات المطلوب إدخالها . وتستخدم قائمة فرعية لكى تساعد المستخدم فى تحديد تكرار إضافة الأرباح .

وبمجرد إدخال البيانات فإن الحسابات المناسبة سوف تجرى ، والإجابة سوف تعرض فى أسفل الشاشة . وستظل الشاشة كما هى دون تغيير حتى يقوم المستخدم بالضغط على أى مفتاح يعود عندئذ إلى القائمة الرئيسية .

المخطط التمهيدى للبرنامج The Program Outline

لكى نخطط للبرنامج دعنا نقوم بتعريف المتغيرات الآتية :-

P = القيمة الحالية للمبلغ

F = القيمه المستقبلية للمبالغ

A = الإيداع الشهرى المنتظم (أو السداد)

RATE = معدل الفائدة السنوى معبراً عنه كنسبة مئوية

I = معدل الفائدة السنوى معبراً عنه ككسر (لاحظ أن I=0.01*RATE)

M% = عدد فترات الفائدة فى العام (بمعنى M%=12 للمضاعفات الشهرية)

N% = عدد السنوات

CHOICE = متغير يحدد نوع الحسابات التى تجرى (CHOICE سوف يخصص لها القيمة 1 أو 2 أو 3 أو 4)

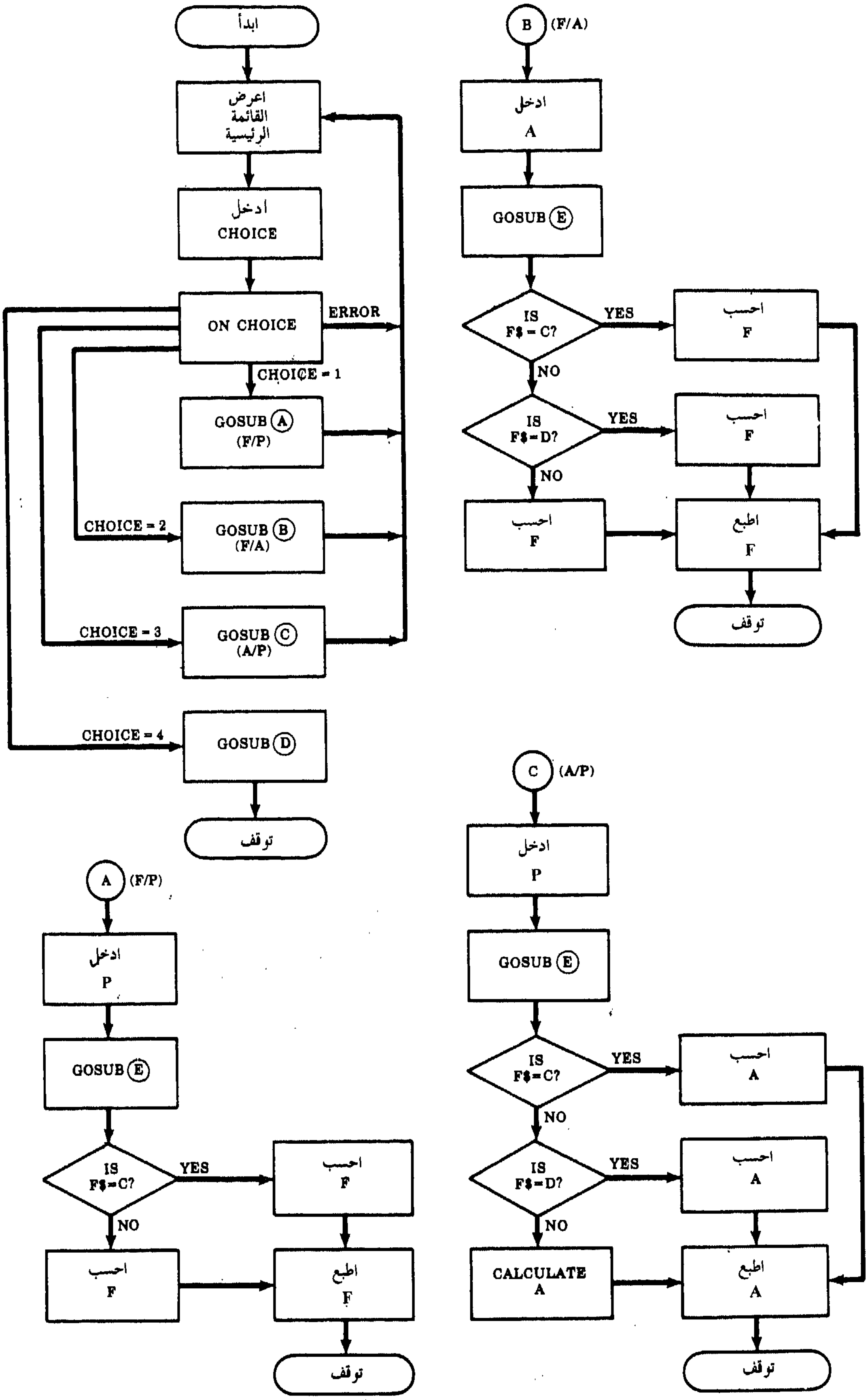
**F$** = متغير يحدد تكرار المضاعفات (F$ سوف يخصص لها "A" أو "S" أو "Q" أو "M" أو "D" أو "C" وذلك للمضاعفات السنوية أو النصف سنوية أو الربع سنوية أو الشهرية أو اليومية أو المستمرة) .

وسوف يعمل البرنامج كما يأتى :ــ

(١) عرض القائمة الرئيسية

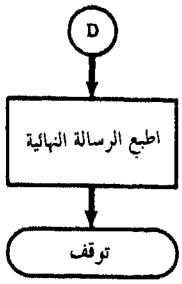
(٢) قراءة قيمة لـ (CHOICE)

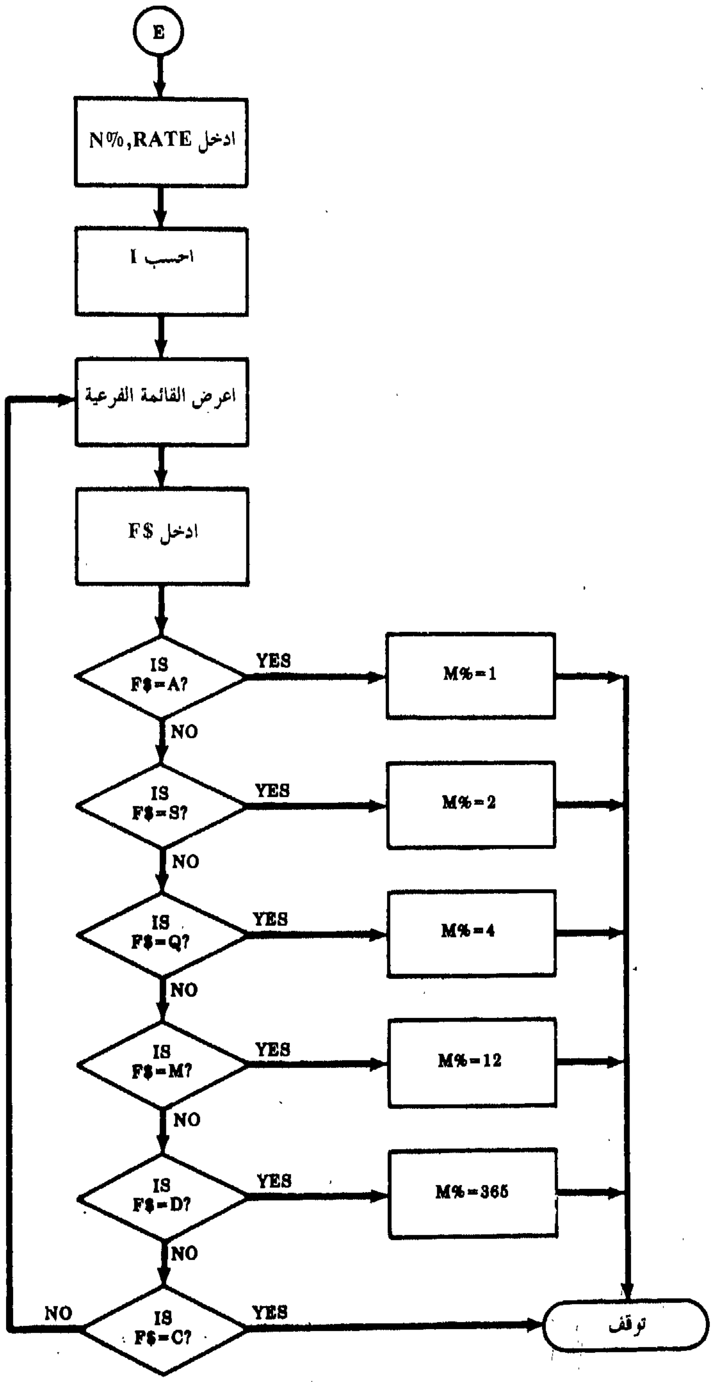
(٣) تفرع لبرنامج فرعى مناسب كما تحدده القيمة المخصصة لـ CHOICE



ابدأ
اعرض القائمة الرئيسية
ادخل CHOICE
ON CHOICE
ERROR
CHOICE = 1
GOSUB (A) (F/P)
CHOICE = 2
GOSUB (B) (F/A)
CHOICE = 3
GOSUB (C) (A/P)
CHOICE = 4
GOSUB (D)
توقف
B (F/A)
ادخل A
GOSUB (E)
IS F$ = C?
YES
احسب F
NO
IS F$ = D?
YES
احسب F
NO
احسب F
اطبع F
توقف
A (F/P)
ادخل P
GOSUB (E)
IS F$ = C?
YES
احسب F
NO
احسب F
اطبع F
توقف
C (A/P)
ادخل P
GOSUB (E)
IS F$ = C?
YES
احسب A
NO
IS F$ = D?
YES
احسب A
NO
CALCULATE A
اطبع A
توقف

شكل ١١ ــ ٤





شكل ١١ ــ ٤ (تكمله)

(أ) CHOICE = 1 (قيمة مستقبلية لمبلغ محدد من المال)

(I) قراءة قيمة P

(II) قراءة قيمة N% وRATE وحساب قيمة لـ I باستخدام المعادلة I = 0.01*RATE

(III) عرض القائمة الفرعية

(IV) قراءة قيمةF$

(V) تخصيص قيمة مناسبة لـ M%

(VI) حساب قيمة F باستخدام المعادلة المناسبة

(VII) عرض قيمة F

(VIII) رجوع للقائمة الرئيسية

(ب) CHOICE = 2 (قيمة مستقبلية لسلسلة من السداد الشهرى)

(I) قراءة قيمة لـ A

(II) قراءة قيم لـ N% وRATE وحساب قيمة لـ I باستخدام المعادلة I=0.01*RATE

(III) عرض القائمة الفرعية

(IV) اقرأ قيمة F$

(V) تخصيص قيمة مناسبة لـ M%

(VI) حساب قيمة F باستخدام المعادلة المناسبة

(VII) عرض قيمة F

(VIII) رجوع للقائمة الرئيسية

(ج) CHOICE =3 (سداد شهرى لقرض)

(I) قراءة قيمة لـ P

(II) قراءة قيمة لـ RATE٪ وحساب قيمة لـ I باستخدام المعادلةI = 0.01*RATE

(III) عرض القائمة الفرعية

(IV) قراءة قيمة لـ F$

(V) تخصيص قيمة مناسبة لـ M%

(VI) حساب قيمة A باستخدام المعادلة المناسبة

(VII) عرض قيمة A

(VIII) رجوع الى القائمة الرئيسية

(د) CHOICE = 4 (END)

اعرض رسالة إشارة توقف وانْهِ الحساب .

لاحظ أن الخطوات (II) و (III) و(IV) و (V) هى نفس الأجزاء ٣(أ) و٣(ب) و٣(جـ) ولذلك فإن التعليمات لهذا الجزء من البرنامج سوف توضع فى برنامج فرعى مستقل ومنفصل . وهذا البرنامج الفرعى سوف يحتوى على توليد القائمة الفرعية .

وشكل ١١ــ ٤ يمثل خريطة تدفق تناظر الخطوط العامة السابقة .

برنامج البيسك The BASIC Program

يبين شكل ١١ــ ٥ برنامج البيسك الكامل . لاحظ أن هذا البرنامج طويل نوعاً ما ، ويرجع ذلك لاستخدام الملقنات واختبارات الخطأ والقوائم . وهذه المظاهر السهلة الاستخدام تبسط استخدام البرنامج مع أنه يحتاج لعمل إضافى من جانب المبرمج .

ويبين الشكلان ١١ــ٦ (أ) و١١ــ٦ (ب) جلسة تفاعل نموذجية . ونرى فى شكل ١١ــ٦ (أ) القائمة الرئيسية مع اختيارات المستخدم (تحتها خط) للاختيار رقم 1 (بمعنى قيمة مستقبلية لمبلغ). الحوار الناتج من هذا الاختيار مبين فى شكل ١١ــ٦ (ب) . لاحظ القائمة الفرعية وعنوانها « تكرار التراكم » والتى تظهر بعد التلقينات للمبلغ الأصلى وعدد السنوات ومعدل الفائدة . ونرى فى النهاية أن الإجابة المحسوبة توجد بالقرب من أسفل الشكل . وهكذا نرى أن الاستثمار الأصلى 1000$ سوف يزداد إلى 3262.04$ إذا سمح بتراكم الفائدة بمعدل 12 فى المئة سنوياً وتتراكم كل ربع سنة لمدة 10 سنوات .

ويوضح الشكلان ١١ــ٧ و١١ــ٨ الحوار الناتج من الاختيارات 2 و3 على الترتيب . والشكلان ١١ــ٧ (أ) و١١ــ٧ (ب) على الأخص يشيران للقيمة المستقبلية لسلسلة من الإيداعات الشهرية . والشكلان١١ــ٨ (أ) و١١ــ٨ (ب) يبينان معدل السداد الشهرى لقرض .. ويبين شكل ١١ــ٩ مايحدث عندما ينتهى البرنامج .

```
10 '****** COMPOUND INTEREST CALCULATIONS ******
20 '
30 KEY OFF: CLS
40 LOCATE 1,1: CLS: PRINT "COMPOUND INTEREST CALCULATIONS"
50 LOCATE 3,1: PRINT "  1 - Future Value of a Given Amount of Money"
60 LOCATE 5,1: PRINT "  2 - Future Value of a Series of Monthly Deposits"
70 LOCATE 7,1: PRINT "  3 - Monthly Loan Repayments"
80 LOCATE 9,1: PRINT "  4 - End"
90 LOCATE 11,1: INPUT "Please enter your selection: ",ANS$
100 CHOICE=VAL(ANS$): IF CHOICE < 1 OR CHOICE > 4 THEN BEEP: LOCATE 11,1:
    PRINT SPACE$(60): GOTO 90
110 ON CHOICE GOSUB 150,270,420,840: GOTO 30
120 '
130 '****** FUTURE VALUE OF A GIVEN AMOUNT OF MONEY (F/P) ******
140 '
150 CLS: LOCATE 1,1: PRINT "FUTURE VALUE OF A GIVEN AMOUNT OF MONEY"
160 LOCATE 3,1: INPUT "Original Amount of Money: $",ANS$
170 P=VAL(ANS$): IF P <= 0 THEN BEEP: LOCATE 3,1: PRINT SPACE$(60): GOTO 160
180 GOSUB 570
190 IF F$="C" OR F$="c" THEN F=P*EXP(I*N%) ELSE F=P*(1+I/M%)^(M%*N%)
200 LOCATE 18,1: PRINT "FINAL AMOUNT =";
210 PRINT USING "$$########,.##";F
220 GOSUB 790
230 RETURN
240 '
250 '****** FUTURE VALUE OF A SERIES OF CONSTANT MONTHLY DEPOSITS (F/A) ******
260 '
270 CLS: LOCATE 1,1: PRINT "FUTURE VALUE OF A SERIES OF MONTHLY DEPOSITS"
280 LOCATE 3,1: INPUT "Amount of Each Payment: $",ANS$
290 A=VAL(ANS$): IF A <= 0 THEN BEEP: LOCATE 3,1: PRINT SPACE$(60): GOTO 280
300 GOSUB 570
310 IF F$="C" OR F$="c" THEN F=A*(EXP(I*N%)-1)/(EXP(I/12)-1): GOTO 350
320 FACTOR=(1+I/M%)^(M%*N%)-1
330 IF F$="D" OR F$="d" THEN F=A*FACTOR/((1+I/M%)^(M%/12)-1): GOTO 350
340 F=(12*A/M%)*FACTOR/(I/M%)
350 LOCATE 18,1: PRINT "FINAL AMOUNT =";
360 PRINT USING "$$########,.##";F
370 GOSUB 790
380 RETURN
390 '
400 '****** MONTHLY LOAN REPAYMENTS (A/P) ******
410 '
420 CLS: LOCATE 1,1: PRINT "MONTHLY LOAN REPAYMENTS"
430 LOCATE 3,1: INPUT "Amount of Money Borrowed: $",ANS$
440 P=VAL(ANS$): IF P <= 0 THEN BEEP: LOCATE 3,1: PRINT SPACE$(60): GOTO 430
450 GOSUB 570
460 IF F$="C" OR F$="c" THEN A=P*(EXP(I*N%)*(EXP(I/12)-1))/(EXP(I*N%)-1):
    GOTO 500
470 FACTOR=(1+I/M%)^(M%*N%)-1
480 IF F$="D" OR F$="d" THEN A=P*(FACTOR+1)*((1+I/M%)^(M%/12)-1)/FACTOR:
    GOTO 500
490 A=(M%*P/12)*(I/M%)*(FACTOR+1)/FACTOR
500 LOCATE 18,1: PRINT "MONTHLY PAYMENT =";
510 PRINT USING "$$#######,.##";A
520 GOSUB 790
530 RETURN
```

شكل ١١ ــ ٥

```
540 '
550 '****** OBTAIN INPUT INFORMATION ******
560 '
570 LOCATE 4,1: INPUT "Number of Years: ",ANS$
580 N%=VAL(ANS$): IF N% <= 0 THEN BEEP: LOCATE 4,1: PRINT SPACE$(60): GOTO 570
590 LOCATE 5,1: INPUT "Annual Interest Rate (Percent): ",ANS$
600 I=.01*VAL(ANS$): IF I <= 0 THEN BEEP: LOCATE 5,1: PRINT SPACE$(60): GOTO 590

610 LOCATE 7,1: PRINT "Frequency of Compounding:"
620 LOCATE 9,1: PRINT "  A - Annual"
630 LOCATE 10,1: PRINT "  S - Semiannual"
640 LOCATE 11,1: PRINT "  Q - Quarterly"
650 LOCATE 12,1: PRINT "  M - Monthly"
660 LOCATE 13,1: PRINT "  D - Daily"
670 LOCATE 14,1: PRINT "  C - Continuous"
680 LOCATE 16,1: INPUT "Please enter your selection: ",F$
690 IF F$="A" OR F$="a" THEN M%=1: RETURN
700 IF F$="S" OR F$="s" THEN M%=2: RETURN
710 IF F$="Q" OR F$="q" THEN M%=4: RETURN
720 IF F$="M" OR F$="m" THEN M%=12: RETURN
730 IF F$="D" OR F$="d" THEN M%=365: RETURN
740 IF F$="C" OR F$="c" THEN RETURN
750 BEEP: LOCATE 16,1: PRINT SPACE$(60): GOTO 680
760 '
770 '****** PAUSE ******
780 '
790 LOCATE 22,1: PRINT "(Press any key to continue)";
800 ANS$=INPUT$(1): RETURN
810 '
820 '****** SIGNOFF ******
830 '
840 LOCATE 13,1: PRINT "GOODBYE, COME AGAIN!"
850 END
```

تابع شكل ١١ــ٥

```
COMPOUND INTEREST CALCULATIONS

   1 - Future Value of a Given Amount of Money

   2 - Future Value of a Series of Monthly Deposits

   3 - Monthly Loan Repayments

   4 - End

Please enter your selection: 1
```

(*a*)

```
FUTURE VALUE OF A GIVEN AMOUNT OF MONEY

Original Amount of Money: $1000
Number of Years: 10
Annual Interest Rate (Percent): 12

Frequency of Compounding:

  A - Annual
  S - Semiannual
  Q - Quarterly
  M - Monthly
  D - Daily
  C - Continuous

Please enter your selection: Q

FINAL AMOUNT =    $3,262.04

(Press any key to continue)
```

(*b*)

شكل ١١ ــ ٦

```
COMPOUND INTEREST CALCULATIONS

  1 - Future Value of a Given Amount of Money

  2 - Future Value of a Series of Monthly Deposits

  3 - Monthly Loan Repayments

  4 - End

Please enter your selection: 2
```

(*a*)

```
FUTURE VALUE OF A SERIES OF MONTHLY DEPOSITS

Amount of Each Payment: $100
Number of Years: 7
Annual Interest Rate (Percent): 8.5

Frequency of Compounding:

  A - Annual
  S - Semiannual
  Q - Quarterly
  M - Monthly
  D - Daily
  C - Continuous

Please enter your selection: M

FINAL AMOUNT =    $11,424.37

(Press any key to continue)
```

(*b*)

شكل ١١ـ٧

```
COMPOUND INTEREST CALCULATIONS

  1 - Future Value of a Given Amount of Money

  2 - Future Value of a Series of Monthly Deposits

  3 - Monthly Loan Repayments

  4 - End

Please enter your selection: 3
```

(*a*)

شكل ١١ ـ ٨

```
MONTHLY LOAN REPAYMENTS

Amount of Money Borrowed: $5000
Number of Years: 5
Annual Interest Rate (Percent): 12.5

Frequency of Compounding:

  A - Annual
  S - Semiannual
  Q - Quarterly
  M - Monthly
  D - Daily
  C - Continuous

Please enter your selection: C

MONTHLY PAYMENT =      $112.66

(Press any key to continue)
```

(*b*)

تابع شكل ١١ــ٨

```
COMPOUND INTEREST CALCULATIONS

  1 - Future Value of a Given Amount of Money

  2 - Future Value of a Series of Monthly Deposits

  3 - Monthly Loan Repayments

  4 - End

Please enter your selection: 4

GOODBYE, COME AGAIN!
```

شكل ١١ــ٩

١١ ــ ٣ فحص الخطأ ERROR CHECKING

لقد رأينا فى الأقسام السابقة من هذا الفصل العديد من أسئلة فحص الخطأ، وكلها تختص بأخطاء البيانات المدخلة. ومع كل، فإن هناك أنواعاً أخرى من الأخطاء التى يمكن أن تحدث داخل برنامج البيسك . وهذه تحتوى على أخطاء لغوية (على سبيل المثال FOR بدون NEXT ، دليل خارج المدى أو معامل مفقود ... الخ) وأخطاء وقت التشغيل (مثل القسمة على صفر أو قيمة عددية فاقت الحد) وأخطاء الإدخال والإخراج من الأسطوانة الممغنطة (مثل ملف غير موجود أو قرص مملوء) وأخطاء الجهاز (مثل الوسيلة غير متاحة أو انتهاء الورق من آلة الطباعة) .

وتحتوى معظم نسخ البيسك للحاسب الدقيق على جمل خاصة لتصيد الخطأ والتى يمكنها اكتشاف الأخطاء المعينة عندما تحدث ثم تحول التحكم لنظام تصحيح الأخطاء داخل البرنامج . ويتم هذا عادة بواسطة جملة ON ERROR-GOTO (مثل ON ERROR GOTO 200) كما هو موضح فى قسم ٩ــ٢ .

وعادة ما يتم تكويد الأخطاء نفسها فى صيغة عددية (مثل الخطأ رقم 11 قد يكون قسمة على صفر).لذا فإنه من الممكن كتابة جملة IF التى تأخذ إجراء معالجاً بمجرد اكتشاف نوع معين من الخطأ (مثل «IF ERR = 11 THEN PRINT "Division by zero»).

مثال ١١ ــ ٥

فيما يلى جزء من برنامج ميكروسوفت بيسك الذى يدخل رقمين ويحسب الجذر التربيعى للفرق بينهما ويطبع النتيجة . وبرنامج تصيد الخطأ يولد عبارة ثم يكرر برنامج إدخال البيانات إذا كان الفرق بين الأرقام سالباً .

```
200 ON ERROR GOTO 500
210 INPUT "X=";X
220 INPUT "Y=";Y
230 Z=SQR(X-Y)
240 PRINT "Z=";Z
250 ON ERROR GOTO 0
    .
    .
    .
500 IF ERR=5 AND ERL=230 THEN PRINT "Attempt to take square root
    of negative number": PRINT "Please enter data again": PRINT:
    RESUME 210
```

جملة ON ERROR فى سطر 200 تتسبب فى نقل التحكم إلى سطر 500 إذا تم اكتشاف خطأ (أى خطأ) أثناء تنفيذ البرنامج . جملة IF-THEN فى سطر 500 تختبر الخطأ رقم 5 (استدعاء غير سليم) فى سطر 230 . إذا تم اكتشاف خطأ فإنه يتم توليد رسالة خطأ وينتقل للتحكم عائد إلى السطر 210 وعلى ذلك يسمح للمستخدم بإعادة إدخال البيانات .

(لاحظ أن آخر انتقال للتحكم يتم عن طريق جملة RESUME بدلاً من GOTO . وهناك جملة خاصة RESUME يقصد لاستعمالها عند نهاية برنامج تصيد الخطأ (انظر قسم ٩ ــ ٢) .

ولاحظ فى النهاية أن جملة ON ERROR—GOTO فى سطر 250 . هذه الجملة تلغى أى اختبار خطأ تال داخل البرنامج . أو بمعنى آخر فإن جملة ON ERROR GOTO 0 تقضى على فاعلية أى تصيد للخطأ والذى قد تم تعريفه سابقاً .

مثال ١١ ــ ٦ تشغيل درجات امتحانات الطلبة **Processing Student Examination Scores**

يحتوى شكل ١١ ــ ١٠ على برنامج ميكروسوفت بيسك لتشغيل درجات امتحانات الطلبة باستخدام ملف بيانات متتابع . هذا البرنامج مختلف عن البرنامج الموجود فى مثال ٩ ــ ٣٠ (انظر شكل ٩ ــ ١٠) .

ويختلف البرنامج الحالى عن البرنامج السابق فى اعتبارات عديدة . على سبيل المثال يسمح البرنامج الحالى للمستخدم أن يحدد أى اسم للملف الذى يريده . (فى النسخة السابقة كان اسم الملف المطلوب SCORES) . وعلى ذلك فإن السطر 80 يلقن المستخدم عن اسم الملف ، والسطر 350 يسبب تخصيص هذا الاسم للملف المطور الجديد الذى تم تحديثه (UPDATE) عند نهاية البرنامج .

ويحتوى البرنامج الحالى بالإضافة إلى ذلك عدة ملقنات واختبارات الخطأ التى لم تكن موجودة فى النسخة السابقة ، فالسطر 160 على سبيل المثال يبين القيمة المسموح بها لرقم الاختبار ، والسطر 250 يبين القيمة المسموح بها لكل درجة امتحان . وجمل الإدخال هذه مصحوبة باختبارات الخطأ (سطر 170 والسطور 260 و270) بحيث أن القيم المدخلة المستخدمة تقع داخل المدى المبين .

واختبارات الخطأ هى مشابهة لتلك المبينة فى مثال ١١ــ٢ (شكل ١١ــ١) . ومع كل فإنه يمكننا الآن استخدام جملة LOCATE CSRLIN-1,1 فى مواضع عديدة . وهذه الجملة تعيد وضع المؤشر (Cursor) لبداية السطر السابق . (لاحظ أن السطور 160 و250 لا تحتوى جمل LOCATE ، وعلى ذلك فإن إعادة وضع المؤشر لابد أن يكون ضمن برامج فحص الخطأ) .

ويوجد أيضاً اختبار لتحديد إذا ما كان ملف البيانات المعين موجود على القـرص المـرن الحالـى (سطر 390) . وإذا ما كان ملف البيانات غير موجود سوف تعرض رسالة خطأ على الشاشة ويتم تلقين المستخدم ثانياً اسم ملف البيانات . لاحظ أن اختبار الخطأ هذا يتم تنشيطه بواسطة جملة ON ERROR (سطر 70) وهو ما يخالف اختبارات الخطأ الأخرى والتى تختبر البيانات المدخلية مباشرة . لاحظ أيضاً أن اختبار الخطأ هذا ينتهى بجملة RESUME والتى تنقل التحكم ثانياً إلى السطر 80 . ولاحظ فى النهاية أن اختبار الخطأ لاسم الملف ينتهى تنشيطه فى سطر 110 بجملة ON ERROR GOTO 0 .

```
10 '****** PROGRAM TO PROCESS STUDENT EXAMINATION SCORES ******
20 '******          USING SEQUENTIAL DATA FILES          ******
30 '
40 KEY OFF: CLS
50 DIM C(15)
60 PRINT "UPDATING STUDENT EXAMINATION SCORES"
70 ON ERROR GOTO 390
80 PRINT: INPUT "File name: ",F$
90 OPEN "I",1,F$
100 OPEN "O",2,"UPDATE"
110 ON ERROR GOTO 0
120 INPUT #1,TITLE$
130 INPUT #1,TERM$
140 PRINT: PRINT "Course title: ";TITLE$;SPC(10);"Term: ";TERM$
150 PRINT #2,TITLE$: PRINT #2,TERM$
160 PRINT: INPUT "Exam number (1-15): ",ANS$: K=VAL(ANS$)
170 IF K < 1 OR K > 15 THEN BEEP: LOCATE CSRLIN-1,1: PRINT SPACE$(40):
    LOCATE CSRLIN-2,1: GOTO 160
180 PRINT: INPUT "Calculate averages (Y/N) ";ANS$
190 '
200 '*** BEGIN LOOP ***
210 '
220 PRINT: INPUT #1,N$
230 FOR I=1 TO K-1: INPUT #1,C(I): NEXT I
240 PRINT N$,
250 INPUT "New score (0-100): ",ANS$
260 IF LEFT$(ANS$,1)="0" THEN C(K)=0: GOTO 280 ELSE C(K)=VAL(ANS$)
270 IF C(K) <= 0 OR C(K) > 100 THEN BEEP: LOCATE CSRLIN-1,1: PRINT SPACE$(40):
    LOCATE CSRLIN-1,1: GOTO 240
280 PRINT #2,N$
290 SUM=0
300 FOR I=1 TO K: PRINT #2,C(I);: SUM=SUM+C(I): NEXT I
310 IF ANS$="N" OR ANS$="n" THEN 350
320 AVG=SUM/K
330 PRINT "Average =";AVG
340 PRINT #2,AVG
350 IF NOT EOF(1) THEN 220 ELSE CLOSE: KILL F$: NAME "update" AS F$: END
360 '
370 '*** ERROR TRAP FOR INPUT FILE NAME ***
380 '
390 IF ERR=53 AND ERL=90 THEN PRINT: PRINT "File does not exist": RESUME 80
400 END
```

شكل ١١ ــ ١٠

ويحتوى البرنامج الحالى أيضاً على عدد من ملاحظات إضافية قليلة وعروض على الشاشة بطريقة أفضل تنظيماً نوعاً ما .

ويبين شكل ١١ــ١١ جلسة تفاعل نموذجية باستخدام نفس مجموعة البيانات المدخلة (درجات امتحانات الطلبة) كما فى مثال ٩ــ٣٠ . وهنا أيضاً نجد أن استجابات المستخدم تحتها خط .

وقد يهمنا ان نقارن جلسة التفاعل الحالية بتلك المبينة فى شكل ٩ــ١١، حيث نلاحظ أن المستخدم قد حدد مبدئياً اسم ملف غير موجود فى الجلسة الحالية (فى السطر الثانى) ، ثم نجد أن البرنامج قد تصيد الخطأ وعندئذ يولد الرسالة «File does not exist» ويلقن ثانياً المستخدم لاسم ملف آخر . والباقى من جلسة التفاعل مشابه للجلسة السابقة فيما عدا التفرقة بين عنوان المقرر والفصل الدراسى والملقنات الممتدة . ومع كل فإن آثار اختبارات الخطأ لرقم الاختبار والدرجات الفردية لا تظهر. ولابد للمستخدم أن يقوم بتشغيل البرنامج الحالى حتى يرى تأثير هذه المظاهر .

```
UPDATING STUDENT EXAMINATION SCORES

File name: score

File does not exist

File name: scores

Course title: Comp Sci 141          Term: Fall 1985

Exam number (1-15): 6

Calculate averages (Y/N) ? y

Adams B F      New score (0-100): 75
Average = 71.66666

Brown P        New score (0-100): 80
Average = 65

Davis R A      New score (0-100): 55
Average = 40

Fisher E K     New score (0-100): 5
Average = 4.166667

Hamilton S P   New score (0-100): 90
Average = 91.66666

Jones J J      New score (0-100): 80
Average = 87.5

Ludwig C W     New score (0-100): 70
Average = 53.33333

Osborne T      New score (0-100): 80
Average = 70

Prince W F     New score (0-100): 100
Average = 82.5

Richards E N   New score (0-100): 70
Average = 55

Smith M C      New score (0-100): 75
Average = 67.5

Thomas B A     New score (0-100): 10
Average = 21.66667

Wolfe H        New score (0-100): 95
Average = 64.16666

Zorba D R      New score (0-100): 95
Average = 78.33334
```

شكل ١١ ــ ١١

١١ـــ٤ تحقق المستخدم USER VERIFICATION

يقع كل فرد أحياناً فى أخطاء عندما يدخل البيانات . ولا تكون عادة نتيجة ذلك خطيرة حيث يمكن أن يقوم المستخدم تشغيل البرنامج ببساطة باستخدام البيانات الخاطئة ، ثم يقوم باعادة إدخال البيانات الصحيحة وتشغيل البرنامج مرة أخرى .

ومع كل ..فأحياناً ما تكون النتائج المترتبة على خطأ البيانات المدخلة خطيرة وخصوصاً فى حالة البرامج التى تحتوى على طرق إدخال بيانات مطولة أو برامج تحتاج إلى وقت تنفيذ كبير أو برامج تقوم بتخزين البيانات فى ملفات البيانات . ويمكن أن تؤدى الحاجة لإعادة إدخال البيانات وإعادة التشغيل بمثل هذه البرامج ضياع الوقت كما تسبب ضيقاً .

ويمكن غالباً التعرف على أخطاء البيانات المدخلة ثم يتم تصحيحها قبل تنفيذ البرنامج ببساطة وذلك بالتحقق من صحة البيانات الحالية . ويتم هذا عادة بواسطة تلقين لبعض مجموعات فرعية لمعالم الإدخال المطلوبة (مثل ما يكفى لشغل كل الشاشة بارتياح) ثم السؤال عما إذا كانت قيم المدخلات صحيحة ، فإذا ما أوضح المستخدم أن البيانات صحيحة ، فإن البرنامج يلقن لأى بيانات إضافية قد تكون مطلوبة أو يستمر فى تشغيل البيانات . أما إذا أوضح المستخدم أن البيانات غير صحيحة ، فإنه يطلب من المستخدم أن يعيد إدخال البيانات ويتحقق من صحتها مرة أخرى . وتستمر هذه الطريقة حتى يتم إدخال كل البيانات المطلوبة ويتم التحقق من صحتها .

مثال ١١ ـــ ٧ تخزين بيانات المعامل Storing Laboratory Data

كتب باحث كيميائى برنامج ميكروسوفت بيسك لتخزين البيانات المأخوذة من عينات عديدة من الفحم فى ملف بيانات عشوائى . والمعلومات التالية لكل عينة من الفحم يلزم إدخالها فى الحاسب .

١ ـــ رقم العينة (عدد صحيح ينحصر بين 9999 ـــ 1) .

٢ ـــ رتبة الفحم (هناك أربعة مدخلات مسموح بها فقط : انتراسيت ، بيتيومينوس ، ليجنيت وبيت . وعلى ذلك فإنه يكفى حرف مفرد مثل A ، B ، L أو P لتصنيف رتبة الفحم) .

٣ ـــ النسبة المئوية « رماد » (الشوائب الغير قابلة للاحتراق)

٤ ـــ النسبة المئوية للكبريت .

٥ ـــ النسبة المئوية للرطوبة .

٦ ـــ القيمة الحرارية (سعر حرارى لكل جرام) .

ولكل عينة فإن طريقة إدخال البيانات تبدأ بالعنوان

```
COAL SAMPLE DATABASE SYSTEM
```

وتظهر فى أعلى الشاشة الحالية ، ثم تظهر ملقنات فردية واحدة فى كل مرة لكل بند من المعلومات المطلوبة . وبعد إدخال كل البيانات يطلب من المستخدم أن يتحقق فيما إذا كانت البيانات صحيحة أو غير صحيحة . (المفروض أن المستخدم سيختبر المعلومات التى أدخلها حالاً ولازالت معروضة على الشاشة) فإن كان إدخال البيانات صحيحاً فإن المستخدم سيستجيب بالضغط إما على Y (حرف عالٍ أو منخفض) أو Return . وسيتسبب هذا فى تخزين البيانات المدخلة فى ملف عشوائى يسمى SAMPLES مع رقم العينة معبراً عنه كرقم السجل . ثم يتحرك البرنامج لعينة الفحم التالية .

أما إذا أدخلت البيانات غير صحيحة ، فإن المستخدم سيستجيب بالضغط على N (أو أى حرف بخلاف Y أو Return) إذا ما طلب منه التحقيق من صحة البيانات ، ثم يتم مسح الشاشة وتظهر الرسالة .

```
Please reenter the data for this sample
```

وبعد ذلك ستتكرر سلسلة التلقينات فتسمح للمستخدم أن يعيد إدخال (والتحقق مرة أخرى) البيانات لنفس العينة . وتستمر هذه الطريقة كلها حتى يتحقق المستخدم من أن البيانات الحالية صحيحة .

ويوضح شكل ١١ ــ ١٢ برنامج البيسك الحقيقى ، ويمحو سطر 50 أى عرض سابق على الشاشة لتعريفات مفتاح الدالة ، ويفتح السطر 60 ملف البيانات العشوائى (لاحظ أن جملة OPEN مكتوبة بشكل مختلف نوعاً ما عن الموجود فى الأمثلة السابقة التى أستخدمت ملفات البيانات العشوائية) . وقد يتم تعريف تركيب كل سجل فى السطر 70 . ثم يتم توليد عنوان الشاشة فى السطر 80 .

وتلقن السطور من 100 إلى 120 رقم العينة . وهذا التلقين يوجه المستخدم لإدخال قيمة 0 لكى ينهى البرنامج والقيمة بين 9999 — 1 فيما عدا ذلك . والسطران 110 و120 يحتويان على برنامج تصيد الخطأ لأى قيم مدخلة التى لا تقع فى هذا المدى .

```
10 '****** COAL SAMPLE DATABASE SYSTEM ******
20 '
30 '**** READ INPUT DATA FOR EACH SAMPLE ****
40 '
50 KEY OFF: CLS
60 OPEN "SAMPLES" AS #1 LEN=20
70 FIELD #1, 1 AS R$, 4 AS A$, 4 AS S$, 4 AS M$, 7 AS C$
80 LOCATE 1,1: PRINT "COAL SAMPLE DATABASE SYSTEM"
90 '
100 LOCATE 3,1: INPUT "Sample number (0 to end, otherwise 1-9999): ",ANS$
110 IF LEFT$(ANS$,1)="0" THEN END ELSE N=VAL(ANS$)
120 IF N <= 0 OR N > 9999 THEN BEEP: LOCATE 3,1: PRINT SPACE$(78): GOTO 100
130 '
140 LOCATE 5,1: INPUT "Rank:  A)nthracite, B)ituminous, L)ignite, P)eat: ",ANS$
150 IF ANS$="A" OR ANS$="a" THEN RANK$="A": GOTO 210
160 IF ANS$="B" OR ANS$="b" THEN RANK$="B": GOTO 210
170 IF ANS$="L" OR ANS$="l" THEN RANK$="L": GOTO 210
180 IF ANS$="P" OR ANS$="p" THEN RANK$="P": GOTO 210
190 BEEP: LOCATE 5,1: PRINT SPACE$(78): GOTO 140
200
210 LOCATE 7,1: INPUT "Percent ash (0-100): ",ANS$
220 IF LEFT$(ANS$,1)="0" THEN A=0: GOTO 250 ELSE A=VAL(ANS$)
230 IF A <= 0 OR A > 100 THEN BEEP: LOCATE 7,1: PRINT SPACE$(78): GOTO 210
240 '
250 LOCATE 8,1: INPUT "Percent sulfur (0-100): ",ANS$
260 IF LEFT$(ANS$,1)="0" THEN S=0: GOTO 290 ELSE S=VAL(ANS$)
270 IF S <= 0 OR S > 100 THEN BEEP: LOCATE 8,1: PRINT SPACE$(78): GOTO 250
280 '
290 LOCATE 9,1: INPUT "Percent moisture (0-100): ",ANS$
300 IF LEFT$(ANS$,1)="0" THEN M=0: GOTO 330 ELSE M=VAL(ANS$)
310 IF M <= 0 OR M > 100 THEN BEEP: LOCATE 9,1: PRINT SPACE$(78): GOTO 290
320 '
330 LOCATE 10,1: INPUT "Heating value (cal/gm): ",ANS$
340 IF LEFT$(ANS$,1)="0" THEN C=0: GOTO 370 ELSE C=VAL(ANS$)
350 IF C <= 0 THEN BEEP: LOCATE 10,1: PRINT SPACE$(78): GOTO 330
360 '
370 LOCATE 12,1: INPUT "Is everything ok? (Y/N) ",ANS$
380 IF ANS$="Y" OR ANS$="y" OR ANS$="" THEN 440
390 CLS: PRINT "Please reenter the data for this sample"
400 GOTO 100
410 '
420 '**** STORE THE DATA ****
430 '
440 LSET R$=RANK$
450 LSET A$=MKS$(A): LSET S$=MKS$(S): LSET M$=MKS$(M): LSET C$=MKS$(C)
460 PUT #1, N
470 CLS: GOTO 80
480 END
```

شكل ١١ ــ ١٢

وتلقن السطور من 140 إلى 190 رتبة الفحم . وحيث أن هناك أربع قيم فقط مسموح بها للإجابة (A ، B ، L أو P) فإن هذا التلقين يظهر على الشاشة فى شكل قائمة مختصرة . ونلاحظ أن المستخدم قد يستجيب باستخدام حروف عالية أو منخفضة .

السطور من 210 إلى 230 تلقن للنسبة المئوية للرماد . ومجموعة الجمل هذه تحتوى على برنامج تصيد الخطأ للقيم العددية التى تقع خارج المدى الموضح من 0 إلى 100 فى المائة . ثم تتولد تلقينات مشابهة للنسبة المئوية للكبريت (السطور 250 إلى 270) والنسبة المئوية للرطوبة (السطور 290 إلى 310) والقيمة الحرارية للفحم بالسعر الحرارى لكل جرام (السطور 330 إلى 350) .

وأخيراً فإن تحقق المستخدم يتولد فى السطر 370 ، فإذا بين المستخدم أن البيانات صحيحة فإن القيم المدخلة يتم تخزينها فى ملف البيانات العشوائى (السطور 440 إلى 460) . ويتم مسح الشاشة ، ويتحول التحكم للسطر 80 مبيناً بذلك طريقة إدخال البيانات للعينة التالية .

أما إذا أوضح المستخدم أن البيانات غير صحيحة ، فإن رسالة «Please reenter» تظهر السطر 390 . ويتم نقل التحكم إلى السطر 100 ونلاحظ أن البيانات المدخلة لا يتم تخزينها إلا إذا تحققت صحتها .

```
COAL SAMPLE DATABASE SYSTEM

Sample number (0 to end, otherwise 1-9999): 1330

Rank: A)nthracite, B)ituminous, L)ignite, P)eat: B

Percent ash (0-100): 14.8
Percent sulfur (0-100): 7.0
Percent moisture (0-100): 4.2
Heating value (cal/gm): 6818

Is everything ok? (Y/N) N
```

(*a*)

```
Please reenter the data for this sample

Sample number (0 to end, otherwise 1-9999): 1330

Rank: A)nthracite, B)ituminous, L)ignite, P)eat: B

Percent ash (0-100): 14.8
Percent sulfur (0-100): 7.0
Percent moisture (0-100): 4.2
Heating value (cal/gm): 6188

Is everything ok? (Y/N) Y
```

(*b*)

شكل ١١ ــ ١٣

ويبين شكل ١١ــ١٣ (أ) طريقة إدخال البيانات لعينة فحم نموذجية.وكالمعتاد فإن استجابات المستخدم تحتها خط . ونلاحظ أن المستخدم أوضح أن هذه القيم غير صحيحة . وعلى ذلك فإن البرنامج سيتجاهل البيانات الحالية وسيقوم بإعادة طريقة إدخال البيانات لهذه العينة .

ويوضح شكل ١١ــ١٣ (ب) إدخال البيانات المصححة.ونلاحظ أن القيمة الحرارية مختلفة عن ما هو فى شكل ١١ــ١٣(أ) . وفى هذه المرة أوضح المستخدم أن البيانات صحيحة مما تسبب عنه تخزين بنود البيانات فى ملف البيانات العشوائى .

أسئلة للمراجعة
Review Questions

١١ ــ ١ لخص الخصائص الأربع للبرمجة السهلة الاستخدام التى تم شرحها فى هذا الفصل . ما هو غرض كل منها ؟

١١ ــ ٢ ما هو نوع المعلومات التى تنقل للمستخدم بالتلقين للبيانات المدخلة ؟

١١ ــ ٣ اشرح استخدام طرق فحص الخطأ فيما يتعلق بملقنات البيانات المدخلة . وما هى الوسيلة التى تجعل إحدى هاتين الخاصيتين مكملة للأخرى ؟

١١ ــ ٤ كيف تختلف القوائم عن الملقنات العادية ؟ ولأى نوع من الحالات تكون القوائم أكثر مناسبة ؟

١١ ــ ٥ ما هى الشروط التى قد تجعل من المرغوب فيه تضمين قائمة رئيسية وواحدة أو أكثر من القوائم الفرعية فى نفس البرنامج ؟ .

١١ ــ ٦ صف فائدة استخدام جملة ON ERROR GOTO وجملة RESUME لإجراء فحص الخطأ . وكيف تختلف طرق فحص الأخطاء هذه عن تلك التى تستخدم فيما يتصل بتلقينات البيانات المدخلة ؟ .

١١ ــ ٧ هل جمل ON ERROR GOTO و RESUME متاحة على الحاسب الدقيق الخاص بك ؟ إذا لم تكن كذلك فهل هناك جمل مشابهة متاحة ؟

١١ ــ ٨ حدد الطريقة المستخدمة لتكويد الأخطاء على الحاسب الدقيق الخاص بك . هل دوال ERR و ERL متاحة ؟

١١ ــ ٩ صف الطريقة التى يتحقق بها المستخدم من صحة البيانات . ولأى نوع من البرامج يكون تحقق المستخدم مفيداً ؟

مسائل تكميلية
Supplementary Problems

المسائل التالية تختص بجمع المعلومات بدلاً من الحل الحقيقى للمسائل . أجب عن الأسئلة باستخدام النسخة الخاصة بك من بيسك الحاسب الدقيق .

١١ ــ ١٠ يصف قسم ١١ ــ ١ نوعاً من فحص الخطأ الذى فيه يتم اختبار كمية عددية مدخلة لتحديد ما إذا كانت تقع داخل المدى المقبول للقيم أم لا . وهذا النوع من فحص الخطأ يستخدم جمل ودوال ميكروسوفت البيسك الآتية

LOCATE	**PRINT**
INPUT (including a prompt)	**SPACE$**
IF-THEN	**GOTO**
BEEP	

حدّد ما إذا كانت كل هذه الجمل متاحة على نسختك من بيسك الحاسب الدقيق ، وهل يمكن تفسيرها بنفس الطريقة الموجودة فى قسم ١١ــ١ ، وإذا لم يكن كذلك فكيف يمكن اجراء هذا النوع من فحص الخطأ باستخدام نسختك من بيسك الحاسب الدقيق ؟

١١ — ١١ يبين قسم ١١ — ٣ كيف يتم توليد قائمة باستخدام جمل ودوال ميكروسوفت بيسك التالية :—

CLS	IF-THEN
LOCATE	BEEP
PRINT	SPACE$
INPUT	GOTO
ANS$	ON-GOTO
VAL	

حدد ما إذا كانت كل هذه الجمل متاحة على نسختك من بيسك الحاسب الدقيق ؟ وما إذا كان يمكن تفسيرها بنفس الطريقة كما هو مبين فى قسم ١١ — ٢ ؟ وإذا لم يكن كذلك فكيف يمكن توليد مثل هذه القوائم باستخدام نسختك من بيسك الحاسب الدقيق .

١١ — ١٢ يصف قسم ١١ — ٣ نوعاً آخر من فحص الخطأ يحتوى على اختبارات خطأ النصوص، وخطأ وقت التشغيل، وخطأ الإدخال/ الإخراج للأقراص وأخطاء الجهاز . وهذا النوع من فحص الأخطاء يستخدم دوال وجمل ميكروسوفت بيسك التالية .

ON ERROR GOTO	IF-THEN
RESUME	PRINT
ERR	ERL

حدد ما إذا كانت كل هذه الجمل متاحة على نسختك من بيسك الحاسب الدقيق وما إذا كان يتم تفسيرها بنفس الطريقة كما هو فى قسم ١١ — ٣ ، وإذا لم يكن كذلك فكيف يمكن إجراء هذا النوع من فحص الخطأ باستخدام نسختك من بيسك الحاسب الدقيق ؟ كيف يختلف هذا النوع فى فحص الخطأ عن فحص الخطأ الموضح فى قسم ١١ — ١ ؟

مسائل للبرمجة
Programming Problems

١١ — ١٣ عدل البرنامج الموجود فى مثال ٩ — ١١ (لتوليد أعداد فيبوناسي والبحث عن الأعداد الأولية) بحيث يحتوى التلقين على عرض المدى المسموح به على الشاشة ، أى ما هو عدد أعداد فيبوناسى (23—1) . أضف فحص الخطأ الذى يختبر شروط الأخطاء التالية .

(أ) القيم العددية خارج المدى .

(ب) الحروف غير الرقمية .

١١ — ١٤ عدل البرنامج الموجود فى مثال ٩ — ٢٧ (البحث عن نهاية عظمى) بحيث تحتوى التلقينات على عرض المدى المسموح به لكل قيمة مدخلة على الشاشة . أضف فحوص الخطأ المناسبة التى تختبر القيم الرقمية خارج المدى والحروف غير الرقمية .

١١ — ١٥ عدل المولد بيجلاتين الموجود فى مثال ٩ — ٢٨ بحيث يمكنه إما ترجمة الإنجليزية إلى بيجلاتين أو بيجلاتين إلى الإنجليزية، على أن يحتوى على قائمة بسيطة تسمح للمستخدم أن يختار أيهما .

١١ — ١٦ أعد كتابة البرنامج الموجود فى مثال ٤ — ٩ (لحساب الاستهلاك) بحيث يمكن استخدامه مع شاشة عرض TV . ابدأ كل عملية حسابية بقائمة مبيناً بها نوع الحسابات المتاحة ، وولد حوار المدخلات والمخرجات المحسوبة أسفل القائمة ، ثم امسح الشاشة فيما بين العمليات الحسابية وتأكد من تضمين فحص الخطأ لكل البيانات المدخلة بما فى ذلك اختبارات القائمة .

١١ ــ ١٧ أعد كتابة البرنامج الموجود فى مثال ١٠ ــ ١٢ (متعدد اللغات «hello») بحيث يمكن اختيار لغة معينة عن طريق قائمة تقليدية بدلاً من قلم ضوئى مع تضمين فحص الأخطاء لاختيارات القائمة التى أدخلها المستخدم .

١١ ــ ١٨ حل المسألة ٥ ــ ٥٦ (مطابقة البلاد مع عواصمها) مستخدماً قائمة لتحديد ما إذا كانت العاصمة موجودة لبلد معين أو البلد موجودة لعاصمة معينة . استخدم مفاتيح الدالة لإجراء اختيارك من القائمة ، وتأكد من تضمين تلقينات وفحص الخطأ مع اختيارات القائمة .

١١ ــ ١٩ اكتب برنامج البيسك الكامل للحاسب الدقيق لخلق واستخدام ملف بيانات عشوائى يحتوى على بيانات لأسماء وعناوين وأرقام وتليفونات كما هو موضح فى مسألة ٩ ــ ٦٥ (انظر أيضاً المسألتين ٨ ــ ٤٠ و٩ ــ ٦٤) وتضمين إمكانية كل الخصائص التالية :ــ

(أ) أضف سجلاً جديداً (أى اسم جديد ، عنون ورقم تليفون) للملف .

(ب) أوجد وأعرض على الشاشة سجلًا معيناً .

(جـ) احذف سجلًا معيناً .

(د) اطبع قائمة للملف كله .

(هـ) انتهِ من الحسابات .

إستخدم فحصاً ثنائيًا لايجاد سجلات الأفراد (انظر مثال ٨ ــ ١٣)

اجعل البرنامج يولد قائمة تسمح للمستخدم باختيار أحد المظاهر الموضحة عاليه . تأكد من مسح الشاشة ، ثم ارجع للقائمة بعد اجراء كل الملامح المطلوبة مع ضرورة تضمين التلقينات وفحوص الخطأ المناسبة مع كل اختيار للقائمة .

الفصل ١٢

بيانات الحاسب الدقيق
Microcomputer Graphics

تسمح معظم الحاسبات الدقيقة بعرض المعلومات بيانيا أو فى صورة نصوص . وتسمح مثل هذه العروض بتوليد عدة أنواع مختلفة من الأشكال البيانيه والرسومات . وتدعم بعض الحاسبات الدقيقة العروض البيانية المتعدده الألوان . وتسمح بعضها لأنواع معينه من الأشكال البيانيه بان تكون متحركه . والقدره على توليد الألوان والعروض المتحركة والمحسنة بتأثيرات الصوت توفر الأساس لأنواع مختلفة من العاب الحاسب التى أصبحت شائعه فى السنوات الحديثة .

والحقيقة أن كل نسخ بيسك الحاسبات الدقيقة تحتوى الآن على تعليمات خاصة بالبيانات التى تسمح بتكوين العروض البيانية ببساطة . وعلى سبيل المثال توجد هناك تعليمات فردية لتوليد عدد من الأشكال الشائعة مثل النقط ، السطور ، المستطيلات ، الدوائر والقطاعات الناقصة . ويمكن استخدام هذه الأشكال لتكوين عروض بيانية مختلفة ومعقدة يمكن أن تتضمن استخدام اللون والصوت . وتتاح أيضا تعليمات خاصة بتوليد عروض متحركة . هذا فضلا عن أنه يمكن التحكم فى التحريك باستخدام أجهزة إدخال مساعدة مثل عصا التوجيه والمتجول . ويشرح هذا الفصل استخدام تعليمات بيسك الخاصة لعدد يمثل تطبيقات البيانيات .

وكما فى الفصول السابقة ، سنركز اهتمامنا على نسخ ميكروسوفت بيسك المتقدمة (أى BASICA) والتى يتم تنفيذها على حاسب IBM الشخصى . ومع كل .. فإن كثيرا من الحاسبات الدقيقة الأخرى تستخدم ميكروسوفت بيسك ، وعلى ذلك .. فهى تدعم خصائصا مطابقة أو مشابهة لتلك التى سيتم شرحها فى هذا الفصل .

١٢ــ ١ أساسيات البيانيات GRAPHICS FUNDAMENTALS

لقد رأينا بالفعل أن النص المعروض يتكون من كلمات تتكون من حروف فرديه (أى الحروف والرموز .. الخ) . وعلى ذلك فيمكننا تصور الحروف على أنها العناصر الأساسية لعروض النص . وتوجد حالة مشابهة لعروض البيانيات حيث تكون العناصر الأساسية هى نقط صغيره تسمى عناصر الصورة (Pixels) . ويمكن تجميع هذه النقط لتكوين أشكال أكثر تعقيدا وذلك كما فى حاله تجميع الحروف لتكوين كلمات وجمل وفقرات .

ويقاس مستوى التفصيل (أى التحليل) لعروض البيانيات بدلاله أكبر عدد من النقط الأفقية والرأسية والتى يمكن عرضها فى أى وقت واحد . ويمكن تحديد هذه القيم بواسطة أجهزة الحاسب . وتدعم الحاسبات الشخصية العروض البيانية فى 640 (أفقى) × 200 (رأسى) من نقط عناصر الصوره (Pixels) ، أو ربما 720 × 350 نقطة (Pixels)، وتتغير هذه القيم من حاسب الى آخر . ويمكن الحصول على تحليلات أعلى (أى 1024 × 1024 أو 4096 × 4096) باستخدام أجهزة أغلى ثمنا .

ويحدد ايضا جهاز الحاسب أقصى عدد متاح من الألوان . وعادة لا تدعم الحاسبات الشخصية أكثر من 16 لوناً والبعض يدعم عددا أقل . ويمكن الحصول على عدد أكبر من الألوان باستخدام أجهزة أغلى ثمنا . وإذا كانت المستويات المتعددة للتحليل متاحة فإنه كلما ارتفع التحليل قلت الألوان .

وعلى سبيل المثال « يمكن لحاسب IBM الشخصى فى شكلة الأساسى أن يدعم أسلوبين من البيانيات . ويمكن الإشارة إليهما بالتحليل المتوسط (200x320 مع أربعة ألوان) والتحليل العالى (200x640 بالأسود والأبيض) على الترتيب . ويتم اختيار الأسلوب بأستخدام جملة SCREEN . وعلى ذلك فإن بيانات SCREEN 1 تستدعى أسلوب بيانات التحليل المتوسط SCREEN 2 أسلوب التحليل العالى . (تذكر أيضا أن SCREEN 0 تستدعى أسلوب النص كما هو مبين فى قسم ١٠ ــ ٦) .

وإذا اختيرت بيانات التحليل المتوسط باستخدام جملة SCREEN ، فإنه يمكن اختيار الألوان باستخدام جملة COLOR . ويتم تفسير هذه الجملة بشكل مختلف عن الأسلوب الموجود فى النص كما هو مبين فى قسم ١٠ ـــ ٦ . ويمكن أن تحتوى جملة COLOR على وجه الخصوص على معلومتين ، تحدد الأولى منها لون الخلفية (تذكر أن المعلومه الأولى تحدد لون الواجهة الأمامية عند وجودها فى أسلوب النص) . ويمكن هذا أى قيمة صحيحة بين 0 و 15 معطيا بذلك اختيار 16 لوناً مختلف للخلفية . وتكون الألوان الفردية وما يناظرها من معالم رقمية هى :

0	black	8	gray
1	blue	9	light blue
2	green	10	light green
3	cyan	11	light cyan
4	red	12	light red
5	magenta	13	light magenta
6	brown	14	yellow
7	white	15	high intensity white

وتحدد المعلمة الثانية ، فى جمله Color لوحه الألوان Palette فقط . وهى مجموعة من ثلاثة ألوان تعطى بالإضافة إلى لون الخلفية أربعة ألوان يمكن استخدامها فى عرض بيانى فى أى وقت واحد . وهناك فقط لوحتان مختلفتان متاحتان . وهما محددتان بالقيم العديدية 0 و 1 على الترتيب . وعلى ذلك .. فإن Color 0,0 سوف تختار خلفيه سوداء (المعلومه الأولى) ولوحة ألوان صفر (المعلمه الثانيه) بينما أو Color 0,1 سوف تختار خلفيه سوداء ولوحة 1 . وتكون الألوان الفردية فى داخل كل لوحة كالآتى :ـــ

Palette 0		*Palette 1*	
0	background color	0	background color
1	green	1	cyan
2	red	2	magenta
3	brown	3	white

ويتم اختيار لون معين من داخل اللوحة باستخدام إحدى جمل أشكال البيانيات أى PSET لنقطة (Pixel) و LINE لسطر أو مستطيل و CIRCLE لدائرة أو قطع ناقص . وسوف نتحدث بإفاضه أكثر حول اختيار لون فيما بعد فى هذا الفصل .

مثال ١٢ ـــ ١

نفرض أننا نريد توليد عروض بيانية لتحليل متوسط على حاسب IBM الشخصى بالألوان الأخضر والأحمر والبنى المعروضة على خلفية بيضاء . وعلى ذلك فلابد أن يحتوى برنامج بيسك على التعليمات التالية :ـــ

```
10 SCREEN 1 : COLOR 7,0
```

وتحدد التعليمه الأولى (SCREEN 1) بيانات التحليل المتوسط (أى 320 نقطه أفقيه و 200 نقطه رأسيه) . وتحدد التعليمه الثانية COLOR 7,0 لون أبيض للخلفية ثم تختار لوحة 0 لألوان الواجهة الأمامية .

ومن الجدول السابق نرى أن هذه اللوحة تحتوى على الألوان : الأحمر والأخضر والبنى .

مثال ١٢ ـــ ٢

نعتبر برنامج بيسك الذى سوف يولد عروضاً بيانيه للتحليل العالى على حاسب IBM الشخصى ، فإذا بدأ البرنامج بمسح الشاشة ، فإننا نحتاج للتعليمات الآتية :ـــ

```
10 KEY OFF : CLS
20 SCREEN 2
```

وفى هذه الحالة فإن جملة Color غير مطلوبة ، حيث يمكن توليد العروض البيانيه عاليه التحليل (أى 640 نقطة أفقية و 200 نقطة رأسية) فقط باسود وابيض (أى أشكال بيضاء مقابل خلفيه سوداء) .

١٢ــ٢ النقط والسطور POINTS AND LINES

نسخ ميكروسوفت بيسك التى تستخدم مع حاسب IBM الشخصى تحتوى على جملتى PSET و PRESET اللتان تولدان نقطاً مفردة عند أى موضع معين على الشاشة وبأى لون. ويقصد بأول الجملتين PSET أن تستخدم مع لون يتم اختياره من اللوحة الحالية عندما تكون فى أسلوب بيانيات التحليل المتوسط .

ولاستخدام هذه الجمله فلابد أن يعقب كلمة PSET زوج من المعالم ينحصر داخل قوسين تفصلهما فاصله ، أى (PSET (160, 100) . وتبين هاتان المعلمتان احداثىْ النقطة X ، X . ويمكن فى حاله أسلوب التحليل المتوسط أن تنحصر المعلمه الأولى بين 0 و 319 والمعلمه الثانية من 0 الى 199 . وتمثل النقطة 0 ، 0 أعلى الركن الأيسر للشاشة ، وتمثل النقطة (319,199) أسفل الركن الأيمن .

سيتبع زوج الأحداثيات معلمة ثالثة اختيارية تبين لون النقطة أى 2 و (100 و 160) PSET. وقد تنحصر هذه المعلمه بين 0 و 3 . وفى بيانيات التحليل المتوسط يتم تفسير قيمه هذه المعلمه كأحد الألوان من اللوحه الحاليه النشيطه (أنظر قسم ١٢ ــ ١) . وإذا لم تتضمن جمله PSET صراحة هذه المعلمة الأخيرة فإن اللون رقم 3 سوف يتم اختياره اتوماتيكيا .

مثال ١٢ ــ ٣

يحتوى برنامج ميكروسوفت بيسك مكتوب على حاسب IBM الشخصى على الجمل التالية :

```
10 SCREEN 1 : COLOR 0,1
20 PSET (160,100),2
```

ويحدد السطر 10 بيانيات تحليل متوسط بخلفيه سوداء ولوحه ألوان رقم 1 (الألوان:سماوى وبنفسجى وأبيض) والسطر 20 يتسبب فى توليد نقطة بنفسيجية فى مركز الشاشة .

مثال ١٢ ــ ٤

يحتوى شكل ١٢ ــ ١ على برنامج ميكروسوفت بيسك مكتوب لحاسب IBM الشخصى والذى يولد 100 نقطه مختلفة فى أوضاع عشوائية على الشاشه .

وتستخدم بيانيات تحليل متوسط وتعرض كل نقطة بلون مختار عشوائيا من اللوحة رقم 1 (باستثناء عدم اختيار لون الخلفية حيث أن مثل هذه النقطة تكون غير مرئية) .

```
10 KEY OFF : CLS
20 SCREEN 1 : COLOR 0,1
30 FOR I=1 TO 100
40     X=INT(320*RND)
50     Y=INT(200*RND)
60     CLR=1+INT(3*RND)
70     PSET(X,Y),CLR
80 NEXT I
90 END
```

شكل ١٢ ــ ١

وينقل سطر 10 أيه عروض سابقة لتعاريف مفتاح الدالة ثم يمسح الشاشة . ويحدد سطر 20 بيانيات تحليل متوسط بخلفية سوداء ولوحة الوان رقم 1 .

ويتم توليد النقط الفردية باستخدام الحلقة التكراريه FOR - TO التى تحتوى على السطور 30 الى 80 . ويولد السطران 40 و 50 زوجاً من إحداثيات عشوائية . ويولد سطر 40 رقماً عشوائيًّا تقع قيمته بين 0 و 319 ، ويولد سطر 50 رقماً عشوائيًّا بين 0 و 199 . ويتم اختيار اللون

عشوائيا فى سطر 60 حيث يتم توليد رقم عشوائى تقع قيمته بين 1 و 3 . وفى النهاية فإنه فى الحقيقه يتم توليد كل نقطه فى الحقيقه فى سطر 70 بإستخدام القيم الحاليه التى تم توليدها عشوائيا .

ونشجع القارىء على تنفيذ هذا البرنامج وملاحظة ما يحدث (ولا بد من مشاهدة العروض التى يتم توليدها لبرامج البيانيات حتى يمكن تفهمها) .

اعتبر استخدام جمله PSET فى أسلوب بيانيات التحليل العالى . وقد يتراوح إحداثا x و y من 0 الى 639 ومن 0 الى 199 على الترتيب . وعلى ذلك فالنقطة (0 ، 0) تمثل أعلى الركن الأيسر من الشاشة . والنقطه (639, 199) تمثل أسفل الركن الأيمن .

وسوف تفسر معلمة اللون بشكل مختلف عنه فى اسلوب التحليل المتوسط . فالقيمة الزوجية (أى 0 أو 2) ستبين الأسود ، والقيمة الفردية (أى 1 أو 3) ستبين الأبيض . وإذا كان هذا لا يتضمن معلمة اللون ، فإن اللون رقم 1 (أبيض) سيتم اختياره .

مثال ١٢ ــ ٥

يحتوى برنامج ميكروسوفت بيسك مكتوب لحاسب IBM الشخصى على الجمل الآتية :ــ

```
10 SCREEN 2
20 PSET (320,100)
```

يحدد سطر 10 بيانيات التحليل العالى (ومع لون أبيض على خلفية سوداء) والسطر 20 يسبب توليد نقطة بيضاء عند مركز الشاشة .

ومن الأمور الشيقة أن نقارن هذه الجمل مع مجموعة الجمل المشابهة فى مثال ١٢ــ ٣ لاحظ أن المثال الحالى لا يحتوى على جمله COLOR . لاحظ أيضا أن معلمة اللون لا تتضمنها جملة PSET .

مثال ١٢ ــ ٦

يبين شكل ١٢ ــ ٢ تغييراً فى البرنامج المعروض فى مثال ١٢ ــ ٤ ومع كل .. فإنه يتم توليد 100 نقطة عشوائية فى بيانيات التحليل العالى .

والمنطق أساسا هو نفس ما سبق عرضه ، مع تغيرات صغيرة فى بعض الجمل حتى يلائم اسلوب بيانيات التحليل العالى (لاحظ حذف جملة COLOR فى البرنامج الحالى) .

```
10 KEY OFF : CLS
20 SCREEN 2
30 FOR I=1 TO 100
40     X=INT(640*RND)
50     Y=INT(200*RND)
70     PSET(X,Y)
80 NEXT I
90 END
```

شكل ١٢ ــ ٢

ونشجع القارىء على تنفيذ هذا البرنامج ومقارنة العرض البيانى مع ما تم توليده فى مثال ١٢ ــ ٤ .

دعنا الآن نغير اهتمامنا الى PRESET وهى الجمله الثانيه من ميكرو سوفت بيسك التى تولد نقطه مفرده .وهذه الجملة تطابق PSET فيما عدا تفسير اللون البديل الأفتراضى (أى اللون الذى يتم اختياره تلقائيا إذا كانت معلمة اللون غير موضحة صراحة).وبينما تقوم جملة PSET باختيار اللون رقم 3 تلقائيا بواسطة البديل الأفتراضى ، فإن جملة PRESET سوف تختار تلقائيا لون الخلفية وعلى ذلك فإنه قد يكون مناسبا فى بعض التطبيقات استخدام PSET لتوليد النقط و PRESET لمسح النقط (وذلك بإعادة توليد النقط فى لون الخلفيه) .

مثال ١٢ ــ ٧

اعتبر الجمل التالية التى يتضمنها برنامج ميكروسوفت بيسك مكتوب لحاسب IBM الشخصى .

```
10 SCREEN 1 : COLOR 2,0
     :
100 PSET (160,100),2
110 FOR I=1 TO 2000 : NEXT I
120 PRESET (160,100)
```

يحدد سطر 10 بيانات تحليل متوسط بخلفية خضراء ولوحه ألوان رقم 0 (ألوان أخضر، أحمر وبنى) . ويتسبب سطر 100 فى توليد نقطه حمراء عند مركز الشاشة . ويتم توليد تأخير وقت مختصر بواسطة الحلقة التكرارية الفارغة FOR — TO فى السطر 110 . ويتبع هذا التأخير الوقتى مسح النقطة الحمراء (أى اعادة توليدها فى لون الخلفية) فى سطر 120

مثال ١٢ ـــ ٨ نقط فى الفراغ Dots in Space

نرى فى شكل ١٢ ـــ ٣ برنامج ميكرو سوفت بيسك أكثر شمولا لحاسب IBM الشخصى الذى يُبْنى على بعض الأفكار المعروضة فى الأمثلة السابقة . ويولد هذا البرنامج 200 نقطة مختلفة فى أماكن عشوائية على الشاشة بإستخدام بيانات التحليل المتوسط . ويتم اختيار كل نقطة عشوائيا من اللوحة رقم 1 .

ومع كل تظل النقط مرئية لفترة محدودة من الوقت فقط . وأخيرا فإن كل نقطة تختفى (يتم مسحها) وتستبدل بنقطة جديدة فى موضع آخر على الشاشة . والتأثير الناتج هو شاشة ممتلئة دائما بنقط ملونة ومتباعدة عشوائيا ، إلا أنها تتغير باستمرار كنقط قديمة تختفى وأخرى جديدة تظهر .

```
10 REM *** DOTS ***
20 '
30 SCREEN 1 : COLOR 0,1
40 DIM X(200),Y(200)
50 RANDOMIZE
60 KEY OFF : CLS
70 I=1
80 '
90 '*** begin main loop ***
100 '
110 PRESET(X(I),Y(I))
120 X(I)=INT(320*RND) : Y(I)=INT(200*RND)
130 CLR=1+INT(3*RND)
140 PSET(X(I),Y(I)),CLR
150 I=I+1
160 IF I > 200 THEN I=1
170 GOTO 110
180 END
```

شكل ١٢ ـــ ٣

ودعنا الان نعتبر اساس عمل هذا البرنامج سطرا بسطر . يحدد سطر 30 بيانات تحليل متوسط بخلفية سوداء ولوحة الوان رقم 1 (سماوى ، بنفسجى وابيض) . وجملة DIM فى سطر 40 تؤدى الى تخزين 200 زوج من الإحداثيات (إحداثيات كل النقط يتم عرضها فى أى وقت واحد) . وسطر 50 يستهل مولد الاعداد العشوائية ، وسطر 60 ينقل أى عرض سابق لتعريفات مفتاح الداله ويمسح الشاشة ، ثم تستهل الحلقة التكرارية ذلك فى سطر 70 .

وتحتوى الحلقة التكرارية الرئيسية على السطور 110 إلى 170 . ويتسبب سطر 110 فى مسح النقطة الحالية (أى النقطة رقم 1) . ونحصل بعد ذلك على مجموعة جديدة من الاحداثيات للنقطة رقم 1 فى سطر 130 . ويتسبب سطر 140 فى توليد النقطة الجديدة رقم 1 .

لاحظ أن الإحداثيات الجديدة سوف يتم تخزينها فى المجموعتين المتراصتين X و Y . وهذا يسمح باستدعاء النقطة الحالية ومسحها فى وقت لاحق . ومع كل لا يحدث هذا المسح حتى يتم توليد 199 نقطة اضافية .

وفى النهاية ، يزداد العداد فى سطر 150 ويعاد وضع قيمته إلى 1 إذا كان ذلك ضروريا فى سطر 160 وتسمح هذه الطريقة باستمرار الحلقة التكرارية بغير حدود .

ونهيب بالقارىء ثانيا بتنفيذ هذا البرنامج وملاحظة ما يحدث لكى يكتسب تفهما عاليا للتأثير الديناميكى الذى ينشأ .

وتحتوى نسخة ميكروسوفت بيسك المتاحة لحاسب IMB الشخصى على جملة LINE التى تسمح برسم خط مستقيم بين نقطتين على الشاشة . ويمكن تحديد كل من النقطتين صراحة ، أو يمكن أن تكون إحداهما هى النقطة الأخيرة المشار إليها بواسطة جملة بيانات سابقة .

ويجب أن يتبع كلمة LINE فى أول صورة لها زوجان من المعالم حيث يمثل كل زوج الإحداثين Y,X لإحدى النقط وينحصر الإحداثيان بين قوسين وتفصلهما فاصلة . ويجب فصل النقطتين بواسطة شرطة (علامة ـــ) . أى (150 , 300) -(50 , 20) LINE . وتعتمد قيم الاحداثيات المسموح بها على أسلوب معين للبيانيات (تحليل متوسط أو عالى) كما تم شرحه فى هذا القسم .

ويمكن أن يتبع الإحداثيات معلمة اختيارية تبين لون السطر مثل (300,150) - (20,50) LINE او . والقيمة المخصصة لهذه المعلمه يمكن أن تنحصر بين 3,0 . وتفسر بنفس الطريقة كقيمه معلمه اللون فى جمله PSET .

مثال ١٢ ـــ ٩

يحتوى برنامج ميكروسوفت بيسك مكتوب لحاسب IBM الشخصى على الجمل التالية :

```
10 SCREEN 1 : COLOR 4,1
20 LINE (20,50) – (300,150),3
```

يحدد سطر 10 بيانيات تحليل متوسط بخلفية حمراء ولوحة ألوان رقم 1 (ألوان : سماوى ، وبنفسجى وأبيض) . ويولد سطر 20 قطر أبيض يبدأ فى أعلى اليسار أى النقطة (20,50) إلى أسفل اليمين ، أى النقطة (300,150) .

لاحظ أن المعلمه الأخيرة كان فى الامكان حذفها فى جمله LINE ، أى :

```
20 LINE (20,50) – (300,150)
```

حيث أن اللون رقم 3 يتم اختياره تلقائيا إذا لم تبين القيمة صراحة . وتسمح الصورة الثانية لجملة LINE بحذف الزوج الأول من الاحداثيات ، أى (300,150) — LINE وتسبب هذه الجملة فى رسم خط النقطة التى أشير إليها أخيرا (فى جملة سابقة) للنقطه المعنيه حاليا . وتكون هذه الصورة من جملة LINE مفيدة عند رسم سلسلة من الخطوط المتصلة .

مثال ١٢ ـــ ١٠ السهم المفاجىء A Lightning Bolt

يبين شكل ١٢ ـــ ٤ برنامج ميكروسوفت بيسك قصيراً مكتوباً لحاسب IBM الشخصى الذى يتسبب فى عرض « سهم مفاجىء » أحمر على الشاشه .

```
10 '************ LIGHTNING BOLT ************
20 '
30 KEY OFF : CLS
40 SCREEN 1 : COLOR 0,0
50 LINE (20,20)-(120,80),2 : LINE -(80,80),2
60 LINE -(220,180),2 : LINE -(140,100),2
70 LINE -(190,100),2 : LINE -(110,20),2
80 LINE -(20,20),2
90 END
```

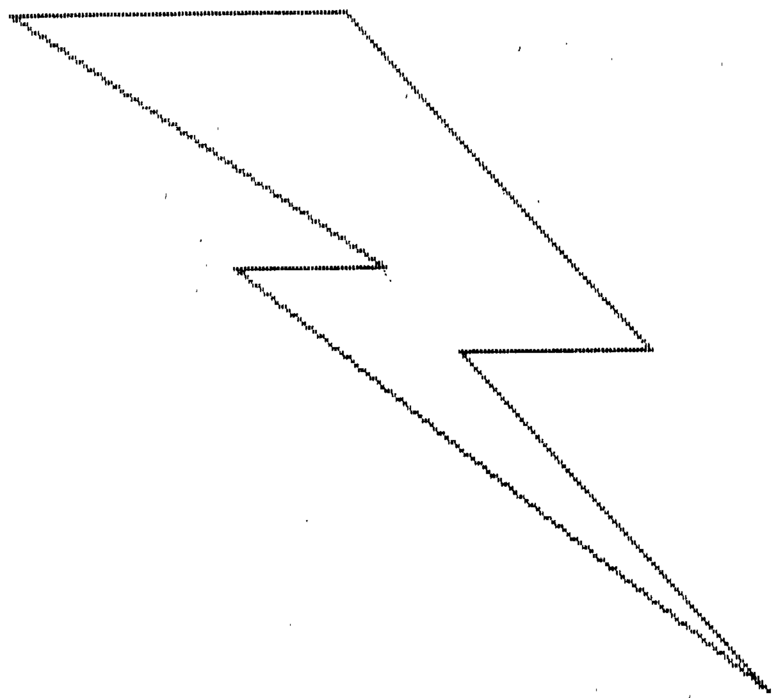
شكل ١٢ ـــ ٤

ويمسح سطر 30 الشاشة ويختار سطر 40 بيانيات التحليل المتوسط بخلفية سوداء وبلوحة الوان رقم 0 (أخضر وأحمر وبنى) . وتحتوى السطور 50 إلى 80 على سبع جمل LINE مولدة بذلك سبعة خطوط متصلة تكون السهم المفاجىء . لاحظ أن كلا من صورتى جملة LINE قد تم استخدامهما . ويجب استخدام الصورة الأولى لتوليد الخط الأول . ومع كل .. فبمجرد إيجاد النقطة الأخيرة تستخدم الصورة الثانية من جملة LINE لتوليد الخطوط الباقية .

وبتنفيذ البرنامج يتم توليد الصوره المبينه فى شكل ١٢ ــ ٥ على الشاشه بخطوط حمراء على خلفيه سوداء . وتدعم بعض الحاسبات الدقيقه صورا اخرى من جمله LINE التى تسمح لمثل هذه الخصائص مثل تحديد خط بالنسبه الى نقطه سابقه بدلا من نقطه مطلقه الى أخرى ، أو توليد خط متقطع . ولن نناقش هذه الخصائص حيث انها تستخدم فى حالات أقل من الخصائص التى سبق شرحها .

وعلى القراء المهتمين الرجوع الى كتيبات برمجه البيسك التى تصاحب حاسباتهم الخاصه لمعلومات اضافيه لهذه الخصائص .

ويمكن استخدام جمله LINE المفرده ايضا لتوليد مستطيل كامل بدلا من خط مفرد وسنرى كيف يتم ذلك فى قسم ١٢ ــ ٣ .



شكل ١٢ ــ ٥

مثال ١٢ ــ ١١ الخطوط المتحركة (الفن الحركى) (Moving Lines (Kinetic Art

دعنا الآن نعيد انتباهنا لبرنامج ميكروسوفت بيسك يعتمد على نفس الفكره العامة للبرنامج المبين فى مثال ١٢ ــ ٨ ، ولكنه يولد عروضا أكثر اثاره .

ونريد توليد سلسلة من الخطوط الملونة فى بيانيات التحليل المتوسط ، موزعة عشوائيا حول الشاشة . ومع كل .. فإن الخطوط المتتابعة لن يتم توليدها مستقلة واحدة عن الأخرى ، بل إن نقطها النهائية المتوالية ستختلف بإزاحات صغيرة . وسيتم عرض مجموعات من الخطوط المتتالية بلون مفرد يتم اختياره عشوائيا من لوحة اللون رقم I . وسيتم تغيير كل من قيمه الإزاحات واللون دوريا كلما استمر البرنامج فى التنفيذ .

وسيقوم البرنامج دائما بعرض نفس عدد الخطوط على الشاشه ولكن سيتم مسح كل خط وتوليد خط جديد .

وعلى ذلك فستظهر الخطوط لتتحرك حول الشاشة عشوائياً ولكن فى نموذج مرتبط بها وبألوان مختلفة . وبالإضافة إلى ذلك ستنشأ أحياناً نماذج إضافية متموجه مشوقه كلما تقاطعت الخطوط الجديده مع الخطوط الموجوده والتاثير النهائى هو أحد المشاهد الجميلة المدهشه .

ويحتوى شكل ١٢ ــ ٦ على البرنامج الحقيقى . وسطر 30 يستدعى بيانات التحليل المتوسط ثم يختار خلفيه سوداء ولوحه ألوان رقم 1 (الألوان سماوى ، بنفسجى وابيض) . ويعرف سطر 40 كل متغيرات البرنامج كمتغيرات من نوع الأعداد الصحيحه . ويقوم سطر 50 بتخزين 150 سطر أى 150 زوج من النقط النهائية . ويبدأ مولد الأرقام العشوائيه فى سطر 60 . ويتم مسح الشاشه فى سطر 70 . ويعطى سطر 80 زوجاً ابتدائيًّا من النقط النهائية (أى (X1,Y1) ، (X2, Y2) على الترتيب) للخط الأول .

ويعطى سطر 90 قيما ابتدائية للعدادات POINTCOUNT, COLORCOUNT, . والأخيرة من هذه ، I ، هى ببساطة عداد حلقة تكرارية يزداد أثناء كل مرور خلال الحلقة التكرارية . وهذا العداد يتراوح بين 1 و 150 ، ويعاد وضعه إلى 1 متى زادت قيمته عن 150 . ويحتاج العدادان الأخيران لبعض الشرح الإضافى .

فمثلا COLORCOUNT هو عداد يحدد متى (ما الذى يمر خلال الحلقة التكرارية) يتم اختيار لون مختلف من اللوحة . ويتم وضع COLORCOUNT بداية عند الصفر فى سطر 90 ، بحيث يمكن توليد قيمة غير صفرية جديدة عشوائيا أثناء المرور الأول خلال الحلقة التكرارية (فى سطر 190) . وهذا يصاحب قيمة جديدة مولدة عشوائيا للون (أيضا فى سطر 190) . وأثناء كل مرور خلال الحلقة التكرارية تتناقص COLORCOUNT بمقدار 1 . وعندما تصبح COLORCOUNT مرة أخرى مساوية لصفر تتولد قيمة جديدة ، ويتم اختيار لون جديد ، وهكذا .

```
10 REM ************ LINES ************
20 '
30 SCREEN 1: COLOR 0,1
40 DEFINT A-Z
50 DIM X1(150),Y1(150),X2(150),Y2(150)
60 RANDOMIZE
70 KEY OFF: CLS
80 X1=120: Y1=70: X2=200: Y2=130
90 COLORCOUNT=0: POINTCOUNT=0: I=1
100 '
110 '************ BEGIN MAIN LOOP ************
120 '
130 '*** erase old line ***
140 '
150 LINE (X1(I),Y1(I))-(X2(I),Y2(I)),0
160 '
170 '*** generate new values for counters if required ***
180 '
190 IF COLORCOUNT=0 THEN COLORCOUNT=5*(1+INT(10*RND)): CLR=1+INT(3*RND)
200 IF POINTCOUNT=0 THEN POINTCOUNT=5*(1+INT(10*RND)): DX1=INT(9*RND)-4:
    DY1=INT(9*RND)-4: DX2=INT(9*RND)-4: DY2=INT(9*RND)-4
210 '
220 '*** generate end points for new line ***
230 '
240 X1=X1+DX1: IF X1 < 0 OR X1 > 319 THEN X1=X1-2*DX1
250 Y1=Y1+DY1: IF Y1 < 0 OR Y1 > 199 THEN Y1=Y1-2*DY1
260 X2=X2+DX2: IF X2 < 0 OR X2 > 319 THEN X2=X2-2*DX2
270 Y2=Y2+DY2: IF Y2 < 0 OR Y2 > 199 THEN Y2=Y2-2*DY2
280 '
290 '*** display new line, then save end points ***
300 '
310 LINE (X1,Y1)-(X2,Y2),CLR
320 X1(I)=X1: Y1(I)=Y1: X2(I)=X2: Y2(I)=Y2
330 '
340 '*** adjust counters and repeat ***
350 '
360 I=I+1
370 IF I > 150 THEN I=1
380 COLORCOUNT=COLORCOUNT-1: POINTCOUNT=POINTCOUNT-1
390 GOTO 150
400 END
```

شكل ١٢ ــ ٦

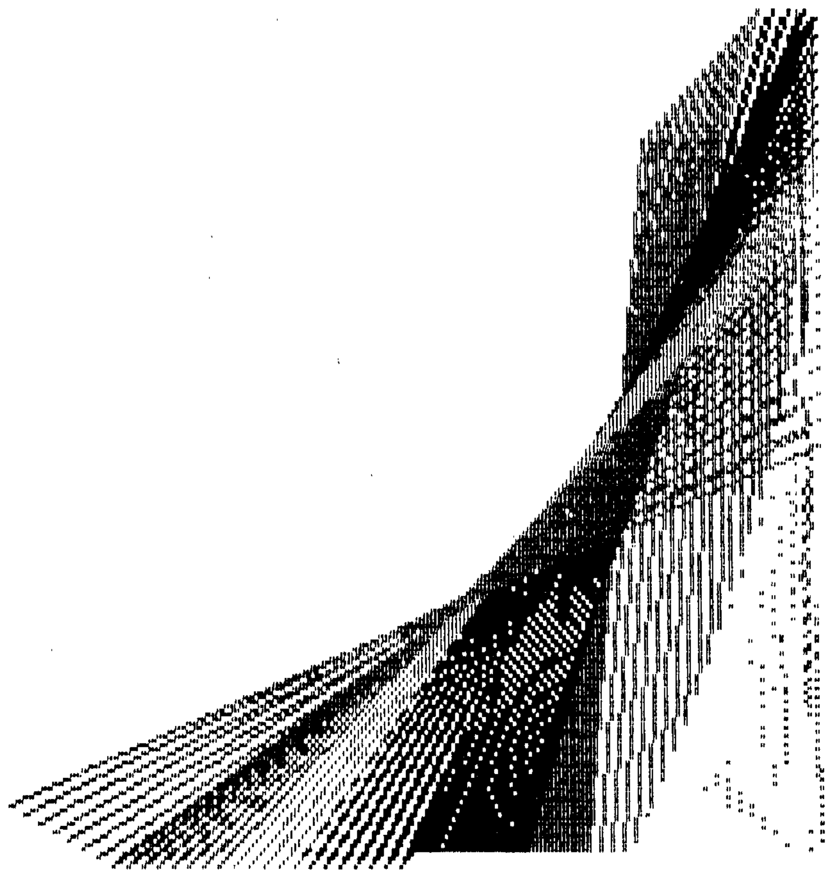
وبالمثل فإن POINTCOUNT هو عداد يحدد متى يتم اختيار قيم مختلفة لتغير إزاحة الخطوط المتتابعة . ويتم وضع POINTCOUNT بداية عند الصفر فى سطر 90 بحيث يتم توليد قيمه غير صفريه أثناء المرور الأول خلال الحلقة التكرارية (فى سطر 200) . هذا بالإضافة إلى أنه سيتم تحديد مجموعة جديدة من القيم لكل من DY2, DX2, DY1. DX1 فى سطر 200 وتحدد هذه المعالم الأربعة الإزاحة للأزواج المتتابعة فى النقط النهائية والتى تحدد بدورها مواقع الخطوط المتتالية . يتم تناقص POINTCOUNT أثناء كل مرور خلال الحلقة التكرارية بمقدار 1 . وعندما تصبح POINTCOUNT مرة أخرى مساوية لصفر ، يتم توليد قيمة جديدة ويتم تحديد مجموعة جديدة من الإزاحات .

ويتراوح الجزء المعاد من البرنامج فى السطور 150 إلى 390 . ويتسبب سطر 150 فى فسح السطر المعرف بمجموعة النقط النهائية التى ترتيبها I والمخزن حاليا فى عناصر الصف (I) X1 ، (I) Y1 ، (I) X2 ، (I) Y2 . ومع كل ، أثناء 150 المرور الأولى خلال الحلقة التكرارية لن يكون هناك مثل هذه النقط النهائية . ويختبر سطر 190 ما إذا كان COLORCOUNT يساوى صفرا . وإذا كان كذلك فإنه يتم اختيار قيمة جديدة عشوائية تتراوح من 5 إلى 50 . وقيمة جديدة للون تتراوح من 1 إلى 3 يتم توليدها عشوائيا .

وسيظل اللون كما هو دون تغير حتى تصبح COLORCOUNT مساوية للصفر مرة أخرى .

ويحتوى سطر 200 على اختبار مماثل POINTCOUNT ، فإذا كانت تساوى صفرا فإنه يتم توليد قيمة جديدة تتراوح من 5 الى 50 عشوائيا . ويتم اختيار مجموعة جديدة من القيم لمعالم الإزاحة . ويتم تخصيص قيمه منفصلة لكل من هذه المعالم ، تتراوح من 4 — الى 4 + . وستظل معالم الإزاحة كما هى دون تغيير حتى تصبح POINTCOUNT مساوية للصفر مرة أخرى .

وتحسب السطور من 240 إلى 270 زوج جديد من النقط النهائية للخط الحالى بالنسبة إلى النقط النهائية للخط السابق . وسيتم التحقق لكل من الإحداثيات الجديدة لتحديد ما إذا كانت قيمتها الجديدة أكبر أو أصغر من اللازم (مسببة بذلك وضع النقط النهائية بعد حواف الشاشة) . وإذا تحقق هذا فيتم ضبط الإحداثى تبعا لذلك .



شكل ١٢ ـــ ٧

ويسبب سطر 310 عرض الخط الجديد على الشاشه ، ويسبب سطر 320 تخزين النقط النهائية لهذا الخط فى صفوف متراصه وهذا يسمح باستدعاء الخط الحالى ومسحه فى زمن لاحق ، ومع كل ، لن يتم هذا المسح حتى يتم توليد 149 سطر إضافى .

وتكمل السطور 390 الحلقه التكراريه وذلك بضبط قيم المعالم ثم العوده الى سطر 150 لبدء مرور آخر . لاحظ أن الحلقه التكراريه سوف تستمر فى التنفيذ بغير حدود حيث أنه لا يوجد شرط معين للإيقاف .

وعند تنفيذ البرنامج يتم توليد نماذج مشابهة لتلك المبينة فى شكل ١٢ ــ ٧ فى عرض يتحرك باستمرار حول الشاشة . وعلى القارىء أن يلاحظ حقيقة البرنامج أثناء تنفيذه على شاشة ملونة حتى يمكنه أن يتفهم كاملا التأثيرات المدهشة التى تتكون .

وقد تم توجيه الأمثلة المعروضة فى هذا الفصل حتى الآن نحو استخدام البيانيات لإحداث تأثيرات مسلية أو فنية . ومع كل .. فإنه يمكن أيضا استخدام البيانيات بفاعلية كبيرة لتوليد عروض مفيدة فى التجارة والتطبيقات الفنية . ويتضح ذلك فى المثال التالى .

مثال ١٢ ــ ١٢ الانحدار الخطى مع العرض البيانى **Linear Regression with Graphical Display**

دعنا الآن ندرس مسألة توفيق خط مستقيم لمجموعة نقط بيانات معطاه بإستخدام طريقة المربعات الصغرى كما هو مبين فى مثال ٧ ــ ٢٢ . (هذه المسأله يشار إليها عاده بالانحدار الخطى*) . وسوف نعرض برنامج ميكروسوفت بيسك كتب لحاسب IBM الشخصى الذى يقوم اولا بالحسابات الرياضية الضرورية ثم يولد عرض بيانى لنقط البيانات الفردية وخط الانحدار . وسوف يحسب مقياس رسم للعرض البيانى حتى يمكن رؤية جميع نقط البيانات وملء الشاشة بأكملها بصرف النظر عن قيم البيانات المعطاه . وسوف نستخدم اسلوب بيانيات التحليل المتوسط لتوليد العرض .

أساسا فإن المسأله هى لتوفيق المعادله الخطيه .

$$y = ax + b$$

لمجموعة من نقط البيانات : (Y1 , X1) ، (Y2 , Y2) ، (Ym , Xm) بتصغير مربعات الاخطاء الى الحد الأقصى كما هو مبين فى مثال ٧ ــ ٢٢ .

وتكون الكميات المجهولة هى قيم المعاملات a و b .وسنحصل على هذه القيم باستخدام الصيغ الرياضيه التالية :ـــ

$$a = (Md_2 - c_1d_1)/(Mc_2 - c_1^2)$$
$$b = (d_1 - ac_1)/M$$

where

$$c_1 = \sum_{i=1}^{M} x_i$$

$$c_2 = \sum_{i=1}^{M} x_i^2$$

$$d_1 = \sum_{i=1}^{M} y_i$$

$$d_2 = \sum_{i=1}^{M} x_iy_i$$

---

* فى الحقيقة ان مصطلح الانحدار الخطى يطبق على توفيق منحى يحتوى عدة أنواع مختلفة من المنحنيات متضمنا دوال أسيه ، دوال لوغاريتميه وكثيرات الحدود (أنظر مثال ٧ ــ ٢٢) .

وعلى ذلك فإن الطريقة الكلية تكون كما يلي :ـــ

١ ـــ قراءة فقط البيانات (Y1 , X1) و (Y2 , X2) ، ... (Ym , Xm) ثم توليد المجاميع التراكميه c_1, c_2, d_1, d_2 كما سبق تعريفها .

٢ ـــ الحل للحصول على المعاملات المجهولة a و b باستخدام الصيغ الرياضيه السابقة الموضحه بعالية .

٣ ـــ تحديد اكبر وأصغر قيمه لكل من X , Y اعدادا للخطوة ٤ التاليه .

٤ ـــ عمل مقياس رسم لنقط البيانات حتى تملأ مساحة عرض البيانيات كاملة .

٥ ـــ توليد العرض الحقيقى للبيانيات فى ثلاث مراحل .

(أ) رسم المحاور .

(ب) رسم نقط البيانيات المفرده (بعد عمل مقياس رسم لها) .

(جـ) رسم خط الانحدار الناتج .

٦ ـــ طباعة معادلة الانحدار المحسوبة فى أعلى الشاشه فوق العرض البيانى .

ويبين شكل ١٢ ـــ ٨ برنامج بيسك الحقيقى . وتمسح السطور من 30 الى 110 الشاشه وتولد عنوان البرنامج وتعرف الصفوف المطلوبة وتعطى قيماً مبدئيه لمفتاح المتغيرات . وتعطى السطور 150 الى 200 برنامج البيانات المدخله . لاحظ ان البرنامج يسمح بادخال عدد غير محدود فى نقط البيانات . وتستمر طريقة ادخال البيانات حتى يضغط المستخدم على مفتاح Return عند تلقينه Y (I) فيدخل بذلك فى سلسلة حروف فارغة .

ويحدد السطران 250,240 الميل (a) والجزء المقطوع من محور y لخط الانحدار المطلوب . ثم يتم فرز بيانات النقط الفردية لتحديد القيمه العظمى والصغرى لكل من Y(1) و X(1) فى السطور 290 الى 350 . ويتم تحديد المقياس الحقيقى لبيانات النقط فى السطور 390 إلى 420 . السطور من 430 الى 450 تولد مقياس النقط النهائية لحساب خط الانحدار .

ويتم العرض البيانى الحقيقى فى السطور من 490 إلى 590 .

ويستدعى سطر 490 بيانيات التحليل المتوسط بخلفيه سوداء ولوحه الوان رقم 1 (الوان سماوى ، بنفسجى وأبيض) . ويتبع هذا أوامر مخرجات البيانيات والتى يتم تجميعها فى ثلاث اقسام مختلفة :ـــ

السطور 500 الى 520 تولد المحاور ، والسطور 540 الى 570 تتسبب فى رسم بيانات النقط المنفردة . والسطر 590 يولد رسم خط الانحدار المحسوب . (لاحظ أن بيانات كل نقطه تنحصر فى مثلث صغير بحيث تكون مرئيه بوضوح على الشاشه وعلى ذلك فإنه يتم استخدام كل من الجملتين PSET ، LINE) .

وفى النهاية يتم عرض المعادله لحساب خط الانحدار وذلك فى أعلى الشاشة . ويولد السطران 630 و 640 هذا العرض . (لاحظ أن ذلك هو عرض للنص رغم أنه قد تم توليده بأسلوب البيانيات) .

```
10 '*************** LINEAR REGRESSION WITH GRAPHICAL DISPLAY *************
20 '
30 KEY OFF: CLS
40 WIDTH 80: SCREEN 0: COLOR 7,0
50 DIM X(100),Y(100)
60 LOCATE 1,20: PRINT STRING$(40,"*")
70 LOCATE 2,20: PRINT "*      LINEAR REGRESSION ROUTINE        *"
80 LOCATE 3,20: PRINT "*";TAB(59);"*"
90 LOCATE 4,20: PRINT "*  Press RETURN after last data point  *"
100 LOCATE 5,20: PRINT STRING$(40,"*"): PRINT
110 I=0: C1=0: C2=0: D1=0: D2=0
120 '
130 '**************** ENTER DATA AND GENERATE COEFFICIENTS **************
140 '
150 I=I+1
160 PRINT "Y(";I;") = ";: INPUT "",ANS$
170 IF ANS$="" THEN M=I-1: GOTO 240 ELSE Y(I)=VAL(ANS$)
180 LOCATE CSRLIN-1,20: PRINT "X(";I;")= ";: INPUT "",X(I)
190 C1=C1+X(I): C2=C2+X(I)^2: D1=D1+Y(I): D2=D2+X(I)*Y(I)
200 GOTO 150
210 '
220 '******************* SOLVE FOR UNKNOWN CONSTANTS *******************
230 '
240 A=(M*D2-C1*D1)/(M*C2-C1^2): IF ABS(A) < 1E-08 THEN A=0
250 B=(D1-A*C1)/M: IF ABS(B) < 1E-08 THEN B=0
260 '
270 '**************** FIND LARGEST AND SMALLEST X AND Y *****************
280 '
290 XMAX=-100000!: YMAX=-100000!: XMIN=100000!: YMIN=100000!
300 FOR I=1 TO M
310     IF X(I) > XMAX THEN XMAX = X(I)
320     IF Y(I) > YMAX THEN YMAX = Y(I)
330     IF X(I) < XMIN THEN XMIN = X(I)
340     IF Y(I) < YMIN THEN YMIN = Y(I)
350 NEXT I
360 '
370 '*********************** SCALE THE X'S AND Y'S ***********************
380 '
390 FOR I=1 TO M
400     X(I)=29+INT(280*(X(I)-XMIN)/(XMAX-XMIN))
410     Y(I)=189-INT(150*(Y(I)-YMIN)/(YMAX-YMIN))
420 NEXT I
430 X1=29: X2=309
440 Y1=189-INT(150*((A*XMIN+B)-YMIN)/(YMAX-YMIN))
450 Y2=189-INT(150*((A*XMAX+B)-YMIN)/(YMAX-YMIN))
460 '
470 '******************** PLOT THE GRAPHICAL DISPLAY *******************
480 '
490 SCREEN 1,1: COLOR 0,1
500 LINE (19,29)-(19,199): LINE -(319,199)
510 FOR IY=59 TO 179 STEP 20: LINE (19,IY)-(23,IY): NEXT IY
520 FOR IX=39 TO 299 STEP 20: LINE (IX,199)-(IX,195): NEXT IX
530 '
540 FOR I=1 TO M
550     PSET(X(I),Y(I)): LINE (X(I),Y(I)-2)-(X(I)-4,Y(I)+2)
560     LINE -(X(I)+4,Y(I)+2): LINE -(X(I),Y(I)-2)
570 NEXT I
580 '
590 LINE (X1,Y1)-(X2,Y2)
600 '
610 '***************** PRINT THE REGRESSION EQUATION ******************
620 '
630 LOCATE 1,5: PRINT "Y =";A;"* X ";
640 IF B >= 0 THEN PRINT "+";B ELSE PRINT "-";ABS(B)
650 END
```

شکل ۱۲ ــ ۸

ويبين الشكلان ١٢ ــ ٩ و ١٢ ــ ١٠ مظهر الشاشة عند تنفيذ البرنامج . ويبين شكل ١٢ ــ ٩ مرحلة ادخال البيانات التى تحدث فى أسلوب النص لمجموعة البيانات الآتية : (تم وضع خط تحت البيانات المدخله فى شكل ١٢ ــ ٩) .

i	y_i	x_i
1	225	10
2	287	20
3	429	30
4	542	40
5	587	50
6	744	60
7	831	70
8	880	80

لاحظ موضع المؤشر الومضى التالى لتلقين Y (9) . ولما كان قد تم ادخال كل البيانات عند هذه النقطه ، فالمستخدم يقوم ببساطه بالضغط على مفتاح Return استجابة للتلقين . وهذا ينهى مرحله ادخال البيانات .

```
                        *****************************************
                        *        LINEAR REGRESSION ROUTINE      *
                        *                                       *
                        *  Press RETURN after last data point   *
                        *****************************************

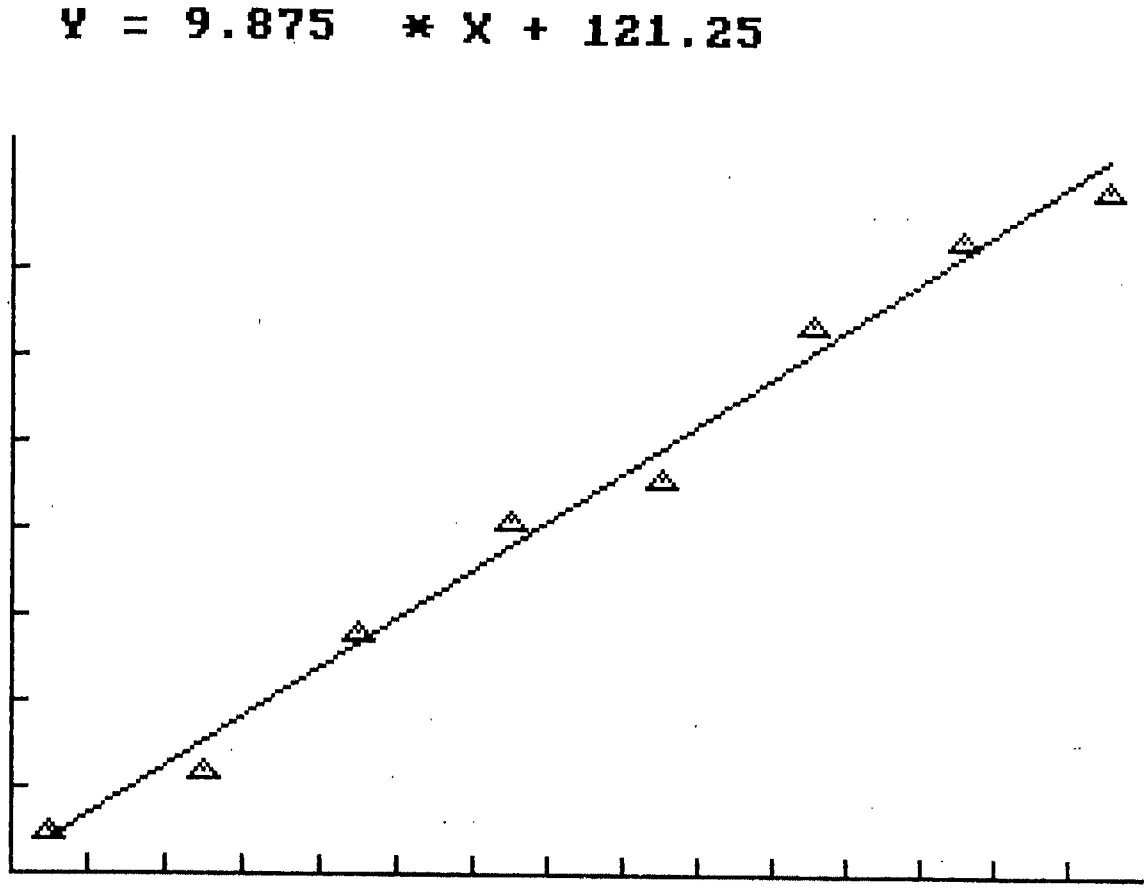
Y( 1 )  = 225           X( 1 )= 10
Y( 2 )  = 287           X( 2 )= 20
Y( 3 )  = 429           X( 3 )= 30
Y( 4 )  = 542           X( 4 )= 40
Y( 5 )  = 587           X( 5 )= 50
Y( 6 )  = 744           X( 6 )= 60
Y( 7 )  = 831           X( 7 )= 70
Y( 8 )  = 880           X( 8 )= 80
Y( 9 )  = _
```

شكل ١٢ ــ ٩

ويبين شكل ١٢ ــ ١٠ (صفحة ٣٧٢) العرض البيانى المناظر والمعادله لحساب خط الانحدار هى :

$$y = 9.875x + 121.25$$

وتقوم هذه المعادلة بتوليد الخط المبين الذى يتضح أنه يمر بنقط البيانات المعطاه رغم أنه قد تم عرض الخط حسب المقياس ليملأ الشاشه . ويبين العرض البيانى الدقه التى يمثل بها خط الانحدار المحسوب لنقط البيانات .



شكل ١٢ ــ ١٠

١٢ ــ ٣ الأشكال SHAPES

تحتوى معظم نسخ بيسك الحاسب الدقيق على جمل تسمح برسم الأشكال البسيطة ، فمثلا يحتوى ميكروسوفت بيسك على صورة خاصة من جملة LINE التى تولد مستطيلات وجملة CIRCLE التى يمكن أن تولد دوائر وقطاعات ناقصة . ويمكن ملء هذه الأشياء بلون متاح إذا رغبنا فى ذلك . ويمكن تكوين تأثيرات بيانية مشوقة بتوفيق هذه الأشياء بطرق مختلفة .

افترض مثلا جملة LINE التى سبق مناقشتها فى القسم الأخير ، فإذا كانت الجملة تنتهى بالحرف B أى : LINE (20,50) - (300,150) ,3,B فإنه يمكن تفسير زوج الإحداثيات على أنهما الركنان المقابلان للمستطيل (أى الصندوق) وعلى ذلك فإنه يمكن رسم المستطيل الذى يكون قطره هو الخط الواصل بين هاتين النقطتين وذلك باللون المبين .

مثال ١٢ ــ ١٣

يحتوى برنامج ميكروسوفت بيسك مكتوب لحاسب IBM الشخصى على الجمل التالية :

```
10 SCREEN 1 : COLOR 4,1
20 LINE (20,50) – (300,150),3,B
```

سطر 10 يحدد بيانيات التحليل المتوسط بخلفية حمراء ولوحة ألوان رقم 1 (ألوان : سماوى ، وبنفسجى وأبيض) . يولد سطر 20 حدود المستطيل الأبيض الذى يصل قطره بين النقطتين (20,50) و (300,150) على خلفية حمراء . (قارن مع مثال ١٢ ــ ٩) .

ويبين المثال التالى برنامج بيسك كاملاً يتم فيه توليد مستطيلات مستخدمة بطريقة أكثر إبداعا .

مثال ١٢ ــ ١٤ المستطيلات المتمدده **Expanding Rectangles**

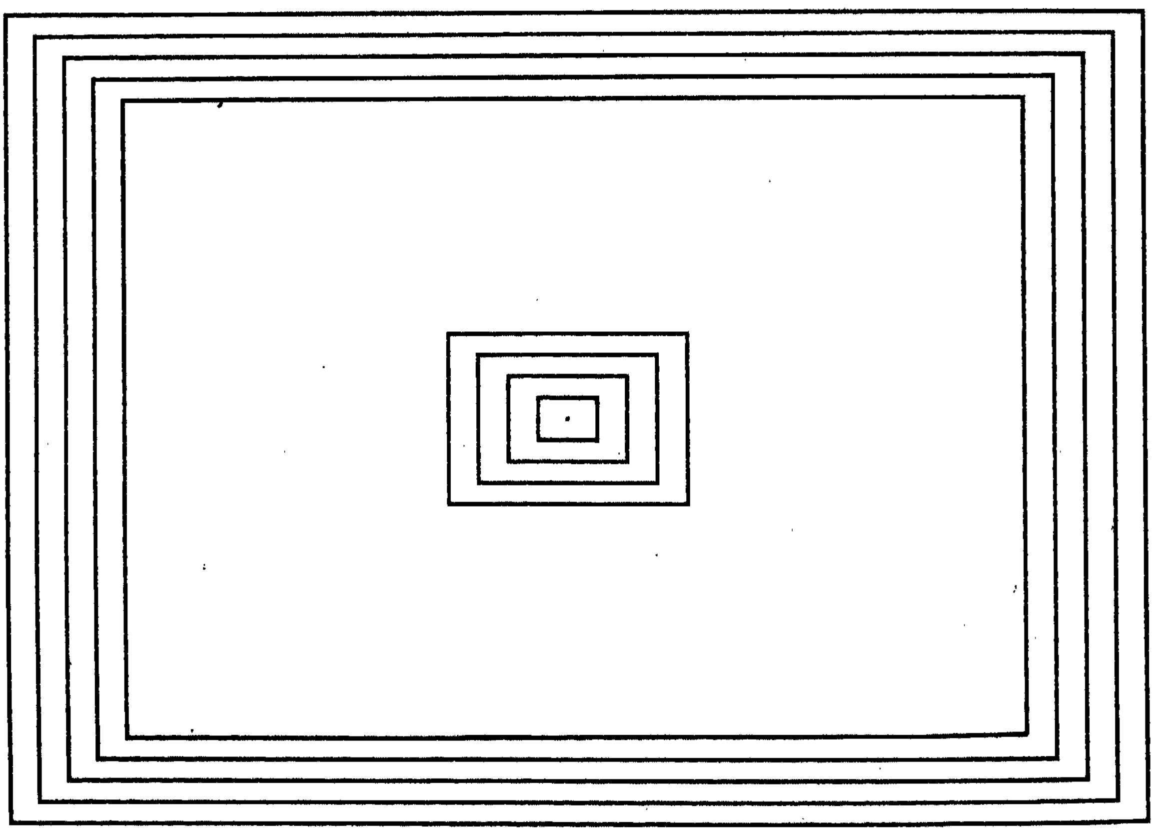
يبين شكل ١٢ ــ ١١ برنامج ميكروسوفت بيسك كاملاً مكتوباً لحاسب IBM الشخصى الذى يتسبب فى تحريك سلسلة من المستطيلات من مركز الشاشة إلى الحواف الخارجية . ويتم توليد مجموعات من المستطيلات بألوان متغيرة للواجهات الأمامية والخلفية مكونة بذلك خداعاً من نبضات لأشكال مستطيلة تنشأ عند مركز الشاشة . ويستخدم البرنامج أيضا الصوت لتحسين التأثيرات البيانية .

```
10 '************ EXPANDING RECTANGLES ************
20 '
30 KEY OFF : CLS
40 SCREEN 1 : COLOR 0,1 : C=3
50 P=0 : Q=-45
60 '
70 '*** BEGIN MAIN LOOP ***
80 '
90 LINE (160-1.6*P,100-P)-(160+1.6*P,100+P),C,B
100 IF Q >= 0 THEN LINE (160-1.6*Q,100-Q)-(160+1.6*Q,100+Q),0,B
110 P=P+5 : IF P > 95 THEN P=0 : C=1+INT(3*RND)
120 Q=Q+5 : IF Q > 95 THEN Q=0
130 IF P=0 OR Q=0 THEN SOUND 20+INT(1000*RND),10
140 GOTO 90
150 END
```

شكل ١٢ ــ ١١

ويقوم سطر 30 فى البرنامج بمسح الشاشة . ويستدعى سطر 40 بيانيات التحليل المتوسط بخلفية سوداء ولوحة ألوان رقم 1 (ألوان : سماوى ، وبنفسجى ، وابيض) ، وبلون واجهة 3 (أبيض) . ويستخدم البرنامج عدادين Q,P يبدآن فى سطر 50 .

وتبدأ الحلقه التكرارية الأساسية بسطر 90 الذى يولد مستطيلا يتحدد حجمه بالقيمة المخصصة إلى P (بداية P=0 مسببه ظهور المستطيل الأول كنقطة عند مركز الشاشة) . ويتحدد لون المستطيل بالقيمة المعينة إلى C (أبيض فى البداية) .



شكل ١٢ ــ ١٢

ويقوم سطر 100 ايضا بتوليد مستطيل بشرط الا تكون Q سالبه . (وتأخذ Q فى البدايه قيمه سالبه حتى يمكن أن ينشأ تأخير وتزداد بعد ذلك وتظل غير سالبه) . ومع كل ، فيتم توليد المستطيل بلون الخلفيه (اسود) .

ويتحدد حجم المستطيل بالقيمه المحدده Q . حيث أن P ، Q سوف تختلفان دائما فى القيمه ، فإن هذا المستطيل لن ينطبق على المستطيل الذى تم توليده فى سطر 90 . وسيكون التأثير هو مسح أحد هذه المستطيلات المرسومة سابقا .

ويقوم السطران 110 ، 120بزيادة P و Q على الترتيب لإعادة وضع كل معلمه إلى الصفر إذا كانت قيمتها تزيد عن 95 . ويتم أيضا توليد لون واجهة جديد عشوائيا عندما يعاد وضع P إلى الصفر .

ويولد سطر 130 صوتا عشوائيا عندما تبدأ سلسلة جديدة من المستطيلات (عندما يعاد وضع Q,P إلى الصفر) ثم يعيد سطر 140 التحكم إلى سطر 90 لمرور آخر خلال الحلقة التكرارية . لاحظ أن الحلقة سوف تستمر فى التنفيذ بغير حدود .

ويبين شكل ١٢ ــ ١٢ نوع العرض البيانى الذى تم توليده بمجرد بدء تنفيذ البرنامج . ومع كل ..فكما هو الحال مع البرامج الأخرى من هذا النوع فإننا نشجع القارىء على تنفيذ هذا البرنامج فعلا حتى يمكنه تفهم التأثيرات الديناميكيه التى تنشأ بالكامل .

نفرض الآن ان جمله LINE تنتهى بالحرفين BF بدلا من الحرف B ، أى LINE (20,50) - (300,150), 3,BF . ويمثل الحرف الاخير املأ الى «FILL» . وتسبب هذه المعلمة المستطيل (الذى تم توليده بالمعلمة B) بأن يمتلىء باللون المعين للواجهة .

مثال ١٢ ــ ١٥

افترض مرة أخرى برنامج ميكروسوفت بيسك مكتوباً لحاسب IBM الشخصى الذى يحتوى على الجمل الآتية

```
10 SCREEN 1 : COLOR 0,0
20 LINE (20,50) - (300,150),1,BF
```

سطر 10 يستدعى بيانيات التحليل المتوسط بخلفيه سوداء ولوحه رقم 0 (الوان اخضر ، احمر ، وبنى) . سطر 20 يتسبب فى توليد مستطيل أخضر مصمت ، يصل قطره بين النقط (20 ، 50) و (125 ، 300) . ويتم عرض هذا المستطيل على خلفيه سوداء (قارن مع مثالى ١٢ ــ ٩ و ١٢ ــ ١٣) .

مثال ١٢ ــ ١٦ المشكال* A Kaleidoscope

أحد التطبيقات المشوقة لاستخدام المستطيلات المليئة هو محاكاة المشكال . وهذا يتضمن توليد قطاعات ملونة وعشوائيه صغيرة (أى مستطيلات) عند أوضاع عشوائيه فى وتد يشغل $\frac{1}{8}$ الشاشة . وتتكرر كل كتلة فى السبع أوتاد الأخرى عند أوضاع متماثلة بالنسبة إلى الوضع

```
10 '************ KALEIDOSCOPE ************
20 '
30 KEY OFF : CLS : SCREEN 1
40 '
50 '*** BEGIN MAIN LOOP ***
60 '
70 U=INT(10*RND) : X=9-INT((U+1)*RND) : Y=9-INT((U+1)*RND) : C=INT(4*RND)
80 LINE (16*X,10*Y)-(16*X+15,10*Y+9),C,BF
90 LINE (304-16*X,10*Y)-(319-16*X,10*Y+9),C,BF
100 LINE (16*X,190-10*Y)-(16*X+15,199-10*Y),C,BF
110 LINE (304-16*X,190-10*Y)-(319-16*X,199-10*Y),C,BF
120 LINE (16*Y,10*X)-(16*Y+15,10*X+9),C,BF
130 LINE (16*Y,190-10*X)-(16*Y+15,199-10*X),C,BF
140 LINE (304-16*Y,10*X)-(319-16*Y,10*X+9),C,BF
150 LINE (304-16*Y,190-10*X)-(319-16*Y,199-10*X),C,BF
160 GOTO 70
170 END
```

شكل ١٢ ــ ١٣

* أداه تحتوى على قطع متحركة من الزجاج الملون وما ان تتغير أوضاعها حتى تعكس مجموعة لانهاية لها من الأشكال الهندسية المختلفة الألوان .

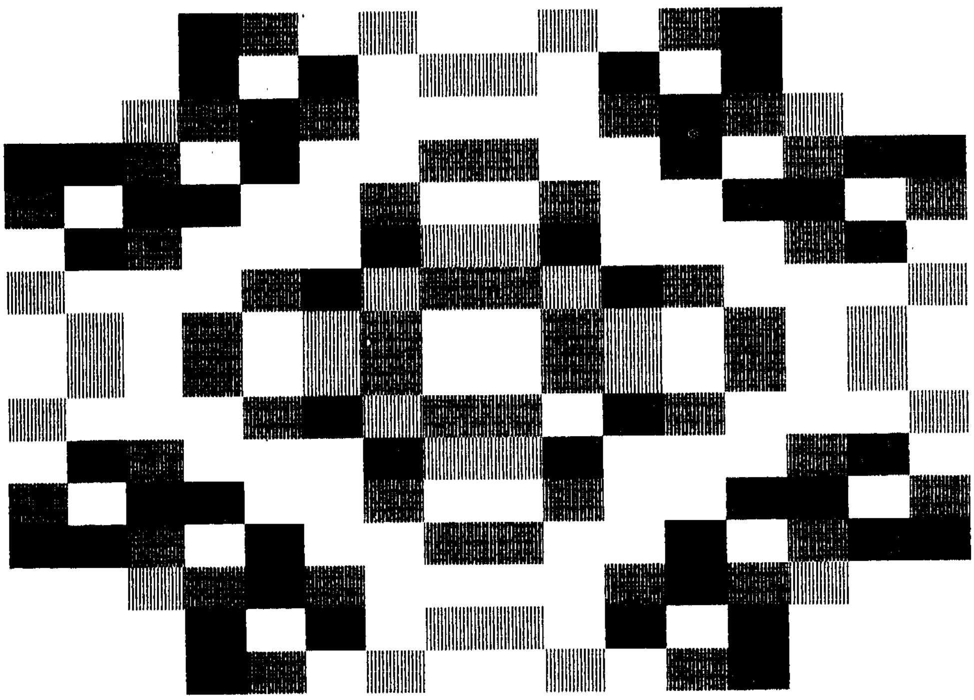
الأصلى . ويمكن جعل المحاكاة أكثر تشويقا بإدخال تحيز ، بحيث يتم توليد معظم الكتل أقرب لمركز الشاشة .

ويحتوى شكل ١٢ ـــ ١٣ على البرنامج الفعلى . ويمسح سطر 30 الشاشة ويستدعى بيانيات التحليل المتوسط . وتكون الخلفية السوداء (لون 0) واللوحة رقم ٥ (ألوان : أخضر ، وأحمر ، وبنى) نشيطتين لغياب البديل الافتراضى .

ويحدث التوليد المتكرر لقطاعات الألوان العشوائية فى الحلقه التكرارية الرئيسية التى تحتوى على السطور 70 إلى 160 . ويولد كل مرور لهذه الحلقه التكرارية ثمانية قطاعات يكون وضعها متماثلاً بالنسبة لكل واحدة منها ، مع وجود كتلة واحدة فى كل وتد . ويكون لكل مجموعة من الكتل نفس اللون ، فعلى ذلك .. فسطر 70 ينشىء وضع الكتلة الأولى ولون كل الكتل الثانى . لاحظ أن قيم الإحداثيين Y,X تكون معتمدة على المعلمة U التى يتم توليدها عشوائيا . وتتسبب هذا فى تجمع أزواج الإحداثيات تجاه مركز الشاشة .

وكل من السطور 80 إلى 150 تولد كتله واحدة . ويكون عرض كل كتلة 16 نقطة «Pixels» وارتفاعها 10 نقط (10 خطوط ممسوحة) . ويتم اختيار النقط النهائية بطريقه تؤدى إلى أن تكون الكتل نموذجا متماثلا على الشاشة . وهذا التماثل هو الذى يحاكى عمل المشكال .

يبين شكل ١٢ ـــ ١٤ نمطاً نموذجيًّا . ومع كل .. نشجع القارىء على أن ينفذ البرنامج فعليا حتى يتفهم ديناميكية النموذج الذى يتكون .



شكل ١٢ ـــ ١٤

تحتوى بعض نسخ ميكرو سوفت بيسك على جمله CIRCLE التى تسمح برسم الدوائر والأقواس والقطاعات الناقصه . وهذه الجمله فى أبسط صورها تحتوى على كلمة CIRCLE يعقبها زوج من الإحداثيات محصور بين قوسين وتفصله فاصله . ولابد أن يعقب الإحداثيات قيمة لنصف القطر اى 80 ,(160,100) CIRCLE .

وقد يظهر رقم صحيح بين اختيار اللون بعد نصف القطر كشىء اختيارى أى 80,3 ,(160,100) CIRCLE . فإذا لم يتحدد اللون بهذه الطريقة فإن لون رقم 3 سيتم اختياره تلقائيا من اللوحة الحالة

مثال ١٢ ــ ١٧

تحتوى الجمل الآتية على برنامج ميكروسوفت بيسك متقدم (BASICA) ، مكتوب لحاسب IBM الشخصى .

```
10 SCREEN 1 : COLOR 4,1
20 CIRCLE (160,100),80,3
```

يحدد سطر 10 بيانات التحليل المتوسط بخلفية حمراء ولوحة ألوان رقم 1 (الألوان : سماوى وبنفسجى وأبيض) . يولد سطر 20 محيط دائرة بيضاء مركزها عند منتصف الشاشة ، أى النقطة (100 و 160) ونصف قطرها 80 نقطة .

مثال ١٢ ــ ١٨ الدوائر المتمددة **Expanding Circles**

يحتوى شكل ١٢ ــ ١٥ على برنامج ميكروسوفت بيسك متقدم ، مكتوب لحاسب IBM الشخصى . يتسبب هذا البرنامج فى توليد مجموعات متكررة من دوائر مركزية (متحددة المركز) ، وتبدأ كل مجموعة منها عند مركز الشاشة وتتحرك قطريا للخارج باتجاه الحافة . سوف ترسم كل مجموعة من الدوائر بلون مفرد يختار عشوائياً . ومع توليد مجموعة متتالية من الدوائر ، فإن الدوائر الجديدة سوف تنشىء نماذج تداخل مع الدوائر السابقة ، مكونة بذلك نماذج شيقة وملونة .

```
10 '****** CIRCLES ******
20 '
30 KEY OFF : CLS
40 SCREEN 1 : COLOR 0,1
50 LINE (39,0)-(279,199),3,B
60 CLR=1
70 '
80 '*** BEGIN LOOP ***
90 '
100 FOR R=5 TO 120 STEP 5
110     CIRCLE (160,100),R,CLR
120 NEXT R
130 CLR=CLR+1 : IF CLR > 3 THEN CLR=1
140 GOTO 100
150 END
```

شكل ١٢ ــ ١٥

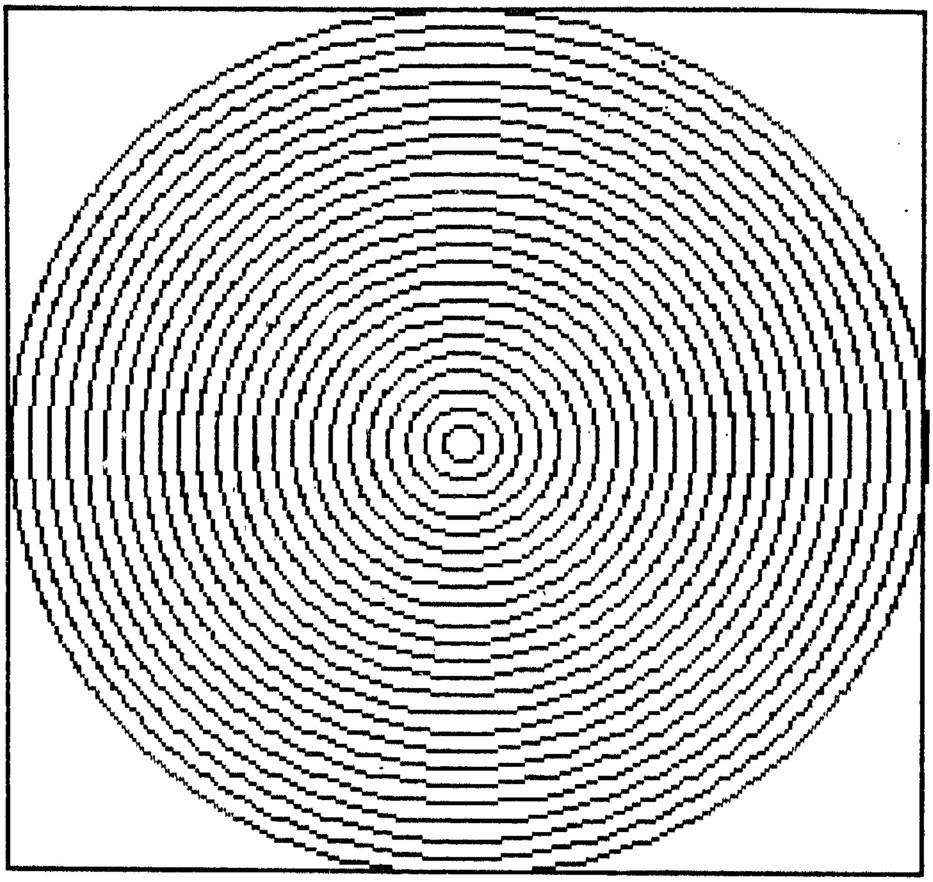
وسنعتبر هذا البرنامج بالتفصيل . ويلغى سطر 30 أى عرض سابق لتعريفات مفتاح الدالة ويمسح الشاشة ، بينما يحدد سطر 40 بيانات التحليل المتوسط ، بخلفية سوداء ولوحة ألوان رقم 1 (ألوان :سماوى، بنفسجى وأبيض) . يولد السطر 50 مربعاً أبيض حول الحواف الخارجية للشاشة . ويتم تحديد لون سماوى لأول مجموعة من الدوائر فى سطر 60 .

ويبدأ الجزء المتكرر من البرنامج بسطر 100 ويستمر حتى سطر 140 . وتُكون السطور 100 و 120 الحلقة التكرارية FOR—TO التى تولد مجموعة من الدوائر المركزية يتحدد لونها بالمعلمه CLR . ثم تتغير قيمة معلمة اللون عندئذ فى سطر 130 . (لاحظ أن قيمة هذه المعلمة سوف تظل دائماً 1 أو 2 أو 3) . ويعيد سطر 140 التحكم إلى الحلقة التكرارية FOR—TO مولداً بذلك مجموعة أخرى من الدوائر .. وهكذا .

يبين شكل ١٢ ــ ١٦ نوع النماذج التى قد يولدها هذا البرنامج . ومع كل فلابد أن تفهم أن هذه النماذج تحتوى على اختلافات مشوقة فى اللون ، وأنها فى حركة ثابتة ، وعلى ذلك يجب أن يلاحظ القارىء البرنامج أثناء التنفيذ حتى يتفهم ماذا يحدث .

ولا تحتوى جملة CIRCLE على ما يؤدى إلى ملء الدائرة بأى لون غير لون الخلفية . ومع كل .. فإن معظم نسخ ميكروسوفت بيسك التى تحتوى جملة CIRCLE تحتوى أيضاً على جملة PAINT ، والتى تسمح لأى شكل مغلق (بما فيه الدوائر والقطاعات الناقصة والمستطيلات) أن يتم مَلْؤه بألوان أخرى . تتكون هذه الجملة من الكلمة PAINT يتبعها زوج من الإحداثيات بين قوسين ويفصلهما فاصلة أى PAINT (160,100) . ويمكن للإحداثين أن يمثلا أى نقطة تنحصر داخل الشكل الذى يراد مَلْؤه .

ويمكن أن يتبع الإحداثيان رقماً صحيحاً ليحدد اللون أى PAINT (160,100),3 وإذا لم يتم تحديد اللون فإنه سيتم اختيار اللون رقم 3 تلقائياً من اللوحة الحالية .



شكل ١٢ ــ ١٦

مثال ١٢ ــ ١٩ السهم الممتلىء المضىء A Filled Lightning Bolt

دعنا مرة أخرى نفحص البرنامج المعطى فى مثال ١٢ ــ ١٠ الذى يولد السهم المضىء (انظر شكل ١٢ ــ ٤) . افترض أننا الآن أضفنا جملة PAINT لهذا البرنامج أى :

90 PAINT (100,40),2

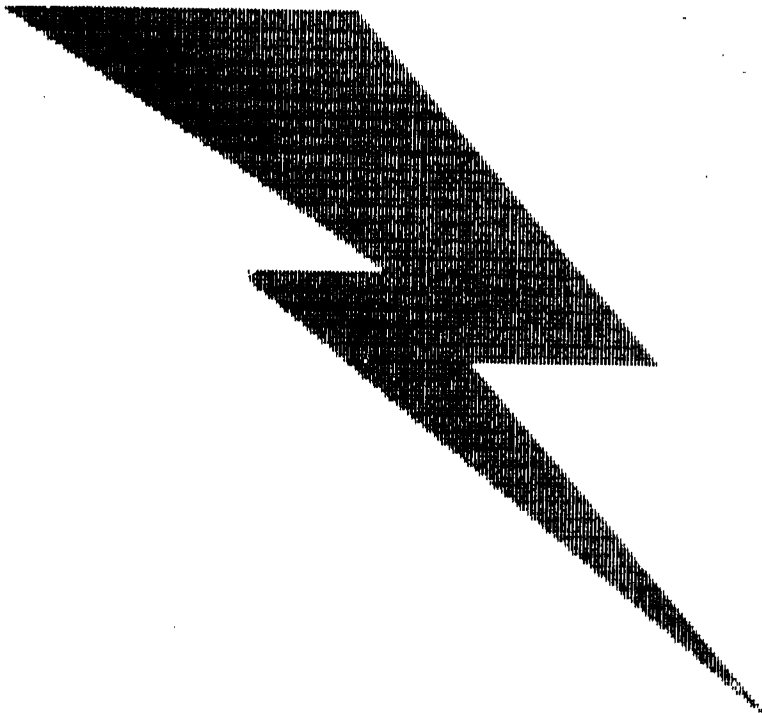
يبين شكل ١٢ ــ ١٧ البرنامج الكامل . لاحظ أن الإحداثيات (100,40) التى تحدد بجملة PAINT تشير إلى النقطة المحصورة داخل الشكل . لاحظ أيضاً أن اللون (2) هو نفس ما تحدده جمل LINE . وعلى ذلك تأثير جملة PAINT هو توليد شكل مصمت أحمر .

```
10 '************ LIGHTNING BOLT ************
20 '
30 KEY OFF : CLS
40 SCREEN 1 : COLOR 0,0
50 LINE (20,20)-(120,80),2 : LINE -(80,80),2
60 LINE -(220,180),2 : LINE -(140,100),2
70 LINE -(190,100),2 : LINE -(110,20),2
80 LINE -(20,20),2
90 PAINT (100,40),2
100 END
```

شكل ١٢ ــ ١٧

ويوضح شكل ١٢ ـــ ١٨ الشكل الذى قد تم توليده بواسطة هذا البرنامج (قارن مع شكل ١٢ ـــ ٥) . تذكر أن الشكل سوف يظهر بالأحمر أمام خلفية سوداء إذا تم تنفيذ البرنامج على حاسب ذى شاشة ملونة .

ويمكن أن يتبع معلمة اللون فى جملة PAINT معلمة إضافية تمثل لون الحدود الجانبية ، أى PAINT (160,100),3,2 ولابد للشكل المطلوب مَلْؤُه أن يتفق مع هذا اللون . وعلى ذلك إذا كانت النقطة التى تحدد بجملة PAINT تكون داخل أكثر من شكل واحد وكل شكل له لون مختلف فإن لون الحدود الجانبية سوف يحدد الشكل المطلوب مَلْؤُه .



شكل ١٢ ـــ ١٨

مثال ١٢ ـــ ٢٠

نفرض أن برنامج ميكروسوفت متقدماً يحتوى على الجمل الأربع الآتية :

```
10 SCREEN 1 : COLOR 4,1
20 CIRCLE (160,100),80,2
30 CIRCLE (160,100),60,3
40 PAINT (160,100),3,2
```

وتسبب هذه الجمل توليد دائرتين مركزيتين . وستكون الدائرة الخارجية (معرفة بسطر 20) بنفسجية بنصف قطر 80 . وستكون الدائرة الداخلية (سطر 30) بيضاء بنصف قطر 60 . وتسبب جملة PAINT فى سطر 40 بأن تملأ الدائرة الخارجية بلون رقم 3 (أبيض) حيث أن لون الحدود فى سطر 40 (أى لون رقم 2) ينطبق على اللون الذى يتحدد فى سطر 20 .

ويمكن استخدام جملة CIRCLE ليس فقط لتوليد الدوائر كاملة ولكن أيضاً أجزاء من الدوائر (أقواس) ولإجراء ذلك فإنه لابد أن يتبع معلمة اللون معلمتين إضافيتين : زاوية البداية وزاوية النهاية . وتقاس هاتان الزاويتان بالطريقة الهندسية التقليدية ، وتتزايد فى اتجاه عكس عقرب الساعة فى النصف الأيمن للمحور الأفقى (أى النصف الأيمن لمحور السينات) . ولابد من التعبير عن كل من الزاويتين بالتقدير الدائرى ، وعلى ذلك فلابد أن تقع فى مدى من 0 إلى 2π أى CIRCLE (160,100),80,1,0,3.14 .

وإذا كانت جملة CIRCLE لا تحتوى على قيمة معلمة اللون ولكنها تحتوى على زاويتى البداية والنهاية ، فإن قيم نصف القطر لابد أن يتبعها فاصلتين أى CIRCLE (160, 100), 80,0,3.14 . وتعبر الفاصلتين المتتاليتين عن معملة لون خالية .

مثال ١٢ ــ ٢١

يحتوى برنامج ميكروسوفت بيسك متقدم (BASICA) مكتوب لحاسب IBM الشخصى على الجمل الآتية :

```
10 SCREEN 1 : COLOR 0,0
20 CIRCLE (160,100),80,1,0,3.14
```

يعين سطر 10 بيانيات التحليل المتوسط بخلفية سوداء ولوحة ألوان رقم 0 (ألوان أخضر ، أحمر وبنى) . ثم يولد سطر 20 النصف الأعلى لدائرة خضراء بنصف قطر 80 . (لاحظ أن زوايتى البداية والنهاية هما π و 0 على الترتيب . إذا استبدلنا سطر 20 مع

```
20 CIRCLE (160,100),80,1,3.14,0
```

فإن النصف الأسفل من دائرة خضراء سوف يتم توليده حيث أن زوايتى البداية والنهاية هما 2π و π على الترتيب . نفس هذا التأثير سوف يحدث بكتابة .

```
20 CIRCLE (160,100),80,1,3.14,6.28
```

الآن افترض أن النسخة العادية من سطر 20 قد تم تبديلها مع

```
20 CIRCLE (160,100),80,,0,3.14
```

سوف تولد هذه الجملة النصف الأعلى لدائرة إلا أنها ستكون باللون البنى بدلاً من الأخضر (حيث أن قيمة معلمة اللون لم تحدد صراحة فإن القيمة البديهية 3 سوف يتم تطبيقها فيظهر اللون البنى) .

يمكن أن تكون زاويتى البداية والنهاية فى جملة CIRCLE سالبة كما يمكن أن تكون موجبة أى CIRCLE (160, 100), 80,1,-3.14 ,-6.28 . وسيظل تفسير الزوايا السالبة كما لو كانت موجبة ، بالمفهوم الهندسى . ومع كل فإن تأثير الزاوية السالبة هو توصيل النقطة النهائية المناظرة بمركز الدائرة .

مثال ١٢ ــ ٢٢

افترض الآن برنامج ميكروسوفت بيسك متقدم والموجود فى المثال الأخير يحتوى على الجمل الآتية :ــ

```
10 SCREEN 1 : COLOR 0,0
20 CIRCLE (160,100),80,1,-3.14,-6.28
```

سوف يولد سطر 20 النصف الأسفل لدائرة خضراء مع خط أفقى يصل النقطتين النهائيتين .

مثال ١٢ ــ ٢٣

الجمل الآتية سوف تولد شكل مألوف لعديد من ألعاب الفيديو السحرية .

```
10 SCREEN 1 : COLOR 0,1
20 PI=3.141593
30 CIRCLE (160,100),40,2,-PI/4,-2*PI
40 CIRCLE (170,78),5,0
50 PAINT (150,100),2,2
```

لاحظ أن الشكل سوف يمتلىء بالبنفسجى .

مثال ١٢ ــ ٢٤ مولد خريطة دائرية A Piechart Generator

الخريطة الدائرية هي شكل دائرى يستخدم عادة لتمثيل البيانات فى شكل نسب مئوية . ويتم تمثيل كل جزء من المعلومات (أى كل نسبة مئوية) بدلالة قطاع دائرى . يتناسب محيط كل قطاع مع قيمة نقط البيانات المناظرة . وهكذا القيمة ٤٠ فى المائة سوف يمثلها قطاع دائرى محيطه هو ٤٠ فى المائة من الدائرة الكلية .

ويبين شكل ١٢ ــ ١٩ مولد بيسك كاملاً لخريطة دائرية ، مكتوباً بميكروسوفت بيسك متقدم لحاسب IBM الشخصى . ومن المفروض أن كل قطعة من البيانات تحتوى على عنوان وقيمة عددية للنسبة المئوية . وعلى ذلك.. يصاحب كل قطاع داخل الخريطة الدائرية عنوان مناسب .

```
10 '************** PIECHART GENERATOR *************
20 '
30 KEY OFF: CLS
40 WIDTH 80: SCREEN 0: COLOR 7,0
50 DIM TITLE$(6),PERCENT(6),A(6)
60 LOCATE 1,20: PRINT STRING$(40,"*")
70 LOCATE 2,20: PRINT "*";TAB(59);"*"
80 LOCATE 3,20: PRINT "*          PIECHART GENERATOR          *"
90 LOCATE 4,20: PRINT "*";TAB(59);"*"
100 LOCATE 5,20: PRINT STRING$(40,"*"): PRINT
110 '
120 '**************** ENTER DATA FOR EACH SECTOR **************
130 '
140 LOCATE 8,1: INPUT "Enter number of sectors (1-6): ",N: PRINT
150 IF N < 1 OR N > 6 THEN BEEP: LOCATE 8,32: PRINT SPACE$(6);: GOTO 140
160 SUM=0
170 FOR I=1 TO N
180     PRINT "Sector";I;SPC(8);
190     INPUT "Title: ",TITLE$(I)
200     LOCATE CSRLIN-1,40: INPUT "Percent: ",PERCENT(I)
210     SUM=SUM+PERCENT(I)
220 NEXT I
230 IF SUM < 99.9 OR SUM > 100.1 THEN BEEP: CLS: LOCATE 23:

    PRINT "Percentages do not sum to 100 - Please try again": GOTO 60
240 '
250 '************ GENERATE THE PIECHART ************
260 '
270 SCREEN 2
280 PI=3.14
290 A1=0
300 FOR I=1 TO N
310     A2=A1+2*PI*PERCENT(I)/100
320     A(I)=(A1+A2)/2
330     CIRCLE (320,100),150,1,-A1,-A2
340     A1=A2
350 NEXT I
360 '
370 '************ LABEL THE SECTORS ************
380 '
390 FOR I=1 TO N
400     C1=(320+75*COS(A(I)))\8+3
410     IF A(I) > PI/2 AND A(I) < 3*PI/2 THEN C1=C1-6
420     R1=(100-30*SIN(A(I)))\8+1
430     IF A(I) < PI THEN R1=R1-1 ELSE R1=R1+1
440     LOCATE R1,C1: PRINT PERCENT(I);"%"
450
460     C2=(320+150*COS(A(I)))\8+5
470     IF A(I) > PI/2 AND A(I) < 3*PI/2 THEN C2=C2-LEN(TITLE$(I))-8
480     R2=(100-62.5*SIN(A(I)))\8+1
490     IF A(I) < PI THEN R2=R2-1 ELSE R2=R2+1
500     LOCATE R2,C2: PRINT TITLE$(I);
510 NEXT I
520 END
```

شكل ١٢ ــ ١٩

وسوف يولد هذا البرنامج الخريطة الدائرية التى تحتوى على ستة قطاعات على الأكثر . ويحتوى البرنامج على أربعة أقسام رئيسية : قسم العنوان والبداية وقسم ادخال البيانات وقسم بيانيات التحليل العالى الذى يولد الخريطة الدائرية الفعلية وقسم الختام الذى يعنون كل من القطاعات .

دعنا نخص التعليمات المفردة ببعض التفصيل . يحذف سطر 30 أى تعريفات لمفاتيح الدالة السابقة ويمسح الشاشة . يحدد سطر 40 أسلوب النص بحروف بيضاء على خلفية سوداء . ويتم تعريف الصفوف المطلوبة للبرنامج فى سطر 50 . ويتم توليد عرض ابتدائى للشاشة فى السطور 60 وحتى 100 .

وتقوم السطور 140 حتى 230 بتكوين قسم إدخال البيانات . ويلقن سطر 140 عدد القطاعات .

ولابد أن تكون هذه القيمة أكبر من الصفر ولكنها لا يمكن أن تزيد عن ستة . يجرى سطر 150 اختبار خطأ للعدد المعين للقطاعات ويعود إلى السطر 140 إذا كانت القيمة المدخلة خارج المدى (حاول ثانية) . ويبدأ المجموع التراكمى لكل النسب فى سطر 160 .

وتلقن السطور 170 حتى 220 لكل بنود البيانات (أولاً عنوان كل قطاع وبعدئذ النسبة المئوية المناظرة) . ويتم تجميع النسب المئوية كما أدخلت . وفى النهاية يختبر سطر 230 ما إذا كان مجموع النسب المئوية كلها يساوى 100 (أو إقريباً منها جداً) . وإذا لم يكن كذلك تتم إعادة توليد العرض الابتدائى للشاشة وتتم إعادة جزء البرنامج الخاص لكل البيانات المدخلة .

ويتم توليد الخريطة الدائرية الفعلية فى سطور 270 إلى 350 . ويتسبب سطر 270 فى تحويل البرنامج إلى بيانيات التحليل العالى ويمسح الشاشة فى هذه العملية . ويحدد السطران 280 و 290 القيم العددية للمتغيرات A1,PI (زاوية البداية للقطاع الأول) . ثم تولد السطور 300 إلى 350 القطاعات المفردة للخريطة الدائرية . وستكون زاوية البداية معروفة دائماً . وعلى ذلك فإنه يتم حساب زاوية النهاية فى سطر 310 ويتم تحديد متوسط زاوية ممثلة فى سطر 320 . ويتم تخزين هذه القيمة الأخيرة فى كتلة متراصة لاستخدامها فيما بعد عند وضع عنوان الخريطة الدائرية . ثم يتم توليد القطاع نفسه فى سطر 330 (لاحظ استخدام جملة CIRCLE بقيم بداية ونهاية سالبة) . وفى النهاية يتم تعيين قيمة زاوية البداية فى سطر 340 للأعداد للقطاع التالى .

تسبب السطور 390 إلى 510 فى عنونة الخريطة الدائرية . لاحظ أن ذلك يتم أثناء بقاء البرنامج فى أسلوب بيانيات التحليل العالى . ويتحدد موضع كل نسبة مئوية فى السطور 400 إلى 440 مستخدمين متوسط الزوايا المحسوبة فى القسم السابق ، ثم يتم عرض النسب المئوية (ولا داعى لمناقشة تفاصيل هذه الجمل المفردة إلا لبيان أن خاصية كل نص لها ارتفاع من 8 نقط ، ولهذا نحتاج إلى عدد صحيح يقبل القسمة على 8) . وبالمثل يتم تحديد موضع كل عنوان ثم يتم عرضه فى سطور 460 إلى 500 . وبذلك فإنه يتم توليد الخريطة الدائرية كاملة بعناوين مناسبة وذلك مع تكملة هذه الحلقة التكرارية . نفرض أن البرنامج المنفذ يستخدم مجموعة البيانات المدخلة التالية. ويبين شكل ١٢ — ٢٠ ديالوج المدخلات (لاحظ أن استجابة المستخدم تحتها خط) . ويبين شكل ١٢ — ٢١ الخريطة الدائرية الناتجة .

Source of Revenue	*Percentage*
Tuition	**45**
State aid	**25**
Research	**15**
Gifts	**8**
Other	**7**

وهناك معلمة واحدة إضافية تصاحب جملة CIRCLE لم تتم مناقشتها وهى معلمة المظهر . وتستخدم لتكون قطاعات ناقصة وأقواس قطاعات ناقصة بدلاً من الدوائر والأقواس الدائرية .

لابد أن تتبع معلمة المظهر زاويتى البداية والنهاية فى جملة CIRCLE أى CIRCLE (160,100),80,1,0,3.14,2 ولابد أن تكون عدداً موجباً (وليس بالضرورة عدداً صحيحاً) أو ما ينتج عنه قيمة عددية موجبة . والقيمة 1 ينتج عنها شكل دائرى أو قريب من ذلك (وقد تختلف القيمة اللازمة للحصول على دائرة تماماً بعض الشىء من حاسب لآخر) . وإذا كانت القيمة أقل من 1 فإنها تولد قطع ناقص أفقى بينما تولد قيمة أكبر من 1 قطع ناقص رأسى، وكلما اختلفت معلمة المظهر عن 1 زاد الاختلاف المركزى .

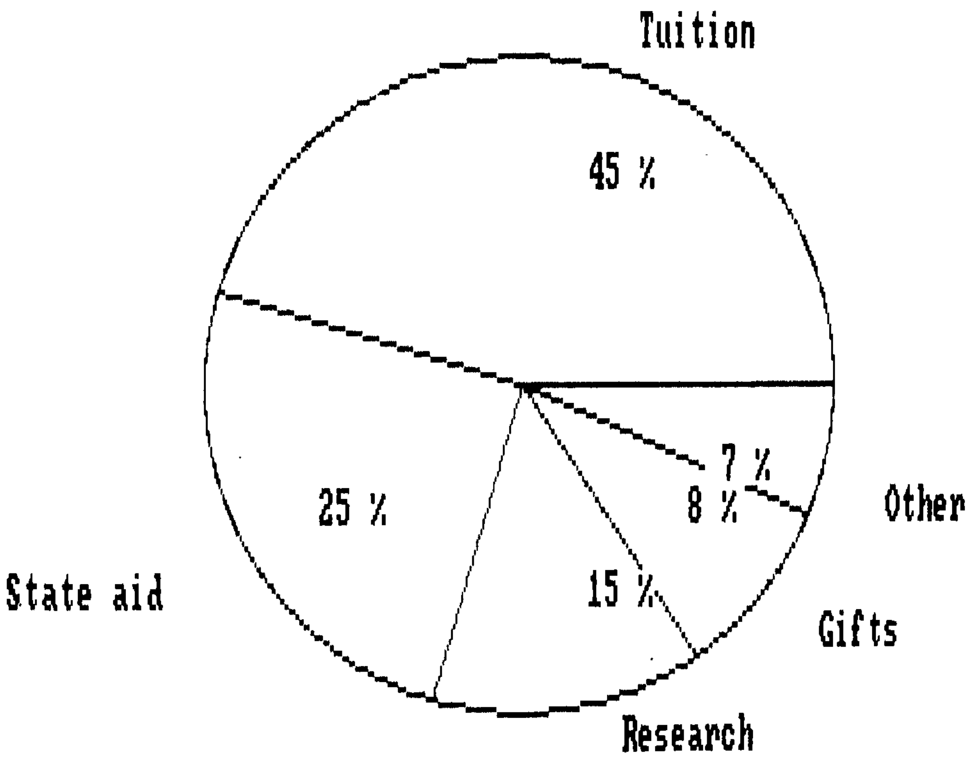
وعندما تستخدم جملة CIRCLE لتوليد قطع ناقص أو قوس ناقص تشير معلمة نصف القطر إلى طول المحور الأكبر .

```
***************************************
*                                     *
*          PIECHART GENERATOR         *
*                                     *
***************************************

Enter number of sectors (1-6): 5

Sector 1          Title: Tuition          Percent: 45
Sector 2          Title: State aid        Percent: 25
Sector 3          Title: Research         Percent: 15
Sector 4          Title: Gifts            Percent: 8
Sector 5          Title: Other            Perpent: 7
```

شكل ١٢ ـــ ٢٠



شكل ١٢ ـــ ٢١

مثال ١٢ ـــ ٢٥

يتضمن برنامج ميكروسوفت بيسك متقدم (BASICA) مكتوب لحاسب IBM الشخصي .

```
10 SCREEN 1 : COLOR 0,0
20 PI=3.141593
30 CIRCLE (160,100),80,1,0,PI,.5
```

سطر 10 يحدد بيانيات التحليل المتوسط بخلفية سوداء ولوحة ألوان رقم 0 (ألوان أخضر ، أحمر وبنى) . ويخصص سطر 20 قيمة للمعلمة PI . ويولد سطر 30 قوس قطع ناقص أخضر يتراوح من 0 إلى π . لاحظ أن القيمة لمعلمة المظهر أقل من 1 . وعلى ذلك فإن المحور الأكبر لقوس القطع الناقص يكون أفقياً . هذا بالإضافة إلى أن طول المحور الأكبر هو 160 نقطة حيث أنه قد تم تخصيص القيمة 80 لنصف القطر .

مثال ١٢ ــ ٢٦

نفرض الآن أن برنامج ميكروسوفت بيسك متقدم يحتوى على الجمل الآتية :—

```
10 SCREEN 1 : COLOR 0,1
20 CIRCLE (160,100),50,,,,.5
```

يحدد سطر 10 بيانيات التحليل المتوسط بخلفية سوداء ولوحة ألوان رقم 1 (ألوان سماوى ، بنفسجى وأبيض) . وسطر 20 يتسبب فى توليد قطع ناقص أفقى أبيض عند مركز الشاشة (لاحظ أن اللون الأبيض هو اللون البديهى المناظر لقيمة 3) .ويكون طول المحور الأكبر هو 100 وحدة حيث أن قيمة نصف القطر هى 50 . وإذا تم تغيير سطر 20 إلى

```
20 CIRCLE (160,100),50,,,,2
```

فإن القطع الناقص يكون رأسياً بدلاً من كونه أفقياً .

مثال ١٢ ــ ٢٧ منطاد بنص متحرك Blimp with Animated Text

يحتوى شكل ١٢ ــ ٢٢ على برنامج ميكروسوفت بيسك متقدم مكتوب لحاسب IBM الشخصى لتوليد رسم منطاد . وعند مركز المنطاد توجد مساحة تعرض فيها رسالة متحركة (وفى هذه الحالةSCHAUM'S OUTLINE) .

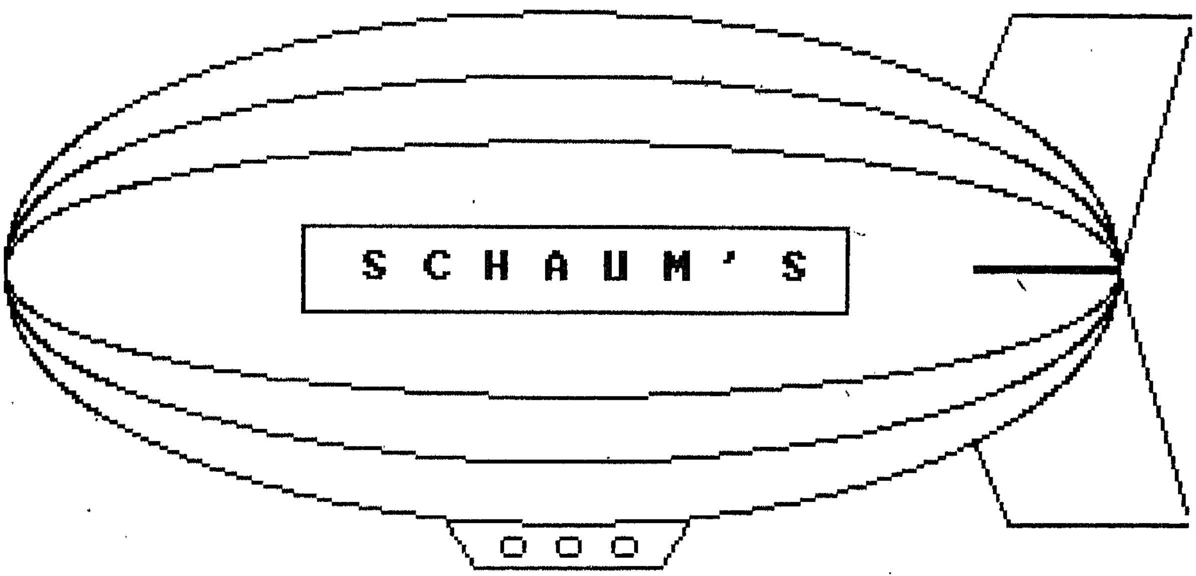
```
10 '************ BLIMP ************
20 '
30 KEY OFF : CLS
40 TEXT$="               S C H A U M ' S   O U T L I N E S  "
50 SCREEN 1 : COLOR 0,1
60 '
70 '*** DRAW THE BLIMP ***
80 '
90 CIRCLE (150,100),150,3,,,.4
100 CIRCLE (150,100),150,3,,,.3
110 CIRCLE (150,100),150,3,,,.2
120 LINE (260,60)-(270,40)
130 LINE -(320,40) : LINE -(300,100)
140 LINE (260,140)-(270,160)
150 LINE -(320,160) : LINE -(300,100)
160 LINE (260,99)-(300,99)
170 LINE (260,100)-(300,100)
180 LINE (118,160)-(125,170)
190 LINE -(175,170) : LINE -(182,160)
200 CIRCLE (135,165),3
210 CIRCLE (150,165),3
220 CIRCLE (165,165),3
230 LINE (80,90)-(225,110),3,B
240 '
250 '*** GENERATE THE MESSAGE ***
260 '
270 FOR I=1 TO 52
280    LOCATE 13,12 : PRINT MID$(TEXT$,I,16);
290    FOR COUNT=1 TO 200 : NEXT COUNT
300    LOCATE 13,12 : PRINT SPACE$(16);
310 NEXT I
320 GOTO 270
330 END
```

شكل ١٢ ــ ٢٢

وتركيب هذا البرنامج هو فى غاية البساطة . سطر 30 يمسح الشاشة وسطر 40 يخصص الرسالة للمتغير TEXT $. ويحدد سطر 50 بيانيات التحليل المتوسط بخلفية سوداء ولوحة ألوان رقم 1 (ألوان سماوى ، بنفسجى وأبيض) . والسطور من 90 إلى 230 تولد المنطاد نفسه . وعلى الأخص فإن السطور 90 ، 100 و 110 ، تولد ثلاثة قطاعات ناقصة مركزية وهى التى تكون الشكل العام للمنطاد . وتولد السطور 120 إلى 170 جزء الذيل . والسطور 180 إلى 230 تولد عجلات الهبوط تحت المنطاد وحدود مساحة العرض .

والسطور 270 إلى 320 تولد الرسالة المتحركة . لاحظ أن مجموعة الجمل هذه تتضمن استخدام دالة MID $ داخل الحلقة التكرارية FOR TO . والفكرة الأساسية هى عرض 16 حرف من النص فى أى وقت واحد . مع كل ، فإن كل مرور خلال الحلقة التكرارية ينتج عنه نقل مجموعة الحروف بحرف واحد إلى اليمين . وكل المجموعة من الحروف يعقبها وقت تأخير قصير ثم مسح منطقة العرض استعداداً لمجموعة الحروف التالية . لاحظ أن سطر 230 يتسبب فى تكرار الحلقة التكرارية ويسبب ذلك استمرار الرسالة المتحركة بغير حدود .

ويبين شكل ١٢ ــ ٢٣ عرضاً نموذجيًا ، وعلى ذلك فلابد للقارىء أن يتذكر أن العرض فى حركة مستمرة عند تنفيذ البرنامج .



شكل ١٢ ــ ٢٣

وتحتوى أيضاً بعض نسخ بيسك الحاسب الدقيق على جملة DRAW التى تسمح بتعريف شكل معقد بدلالة سلسلة من الحروف . فمثلا جملة DRAW «R100 D50 L100 U50 تعرف مستطيل أفقى وذلك بالتحرك يميناً 100 وحدة و 50 وحدة إلى أسفل و 100 وحدة إلى اليسار ثم 50 وحدة إلى أعلى . ويولد تنفيذ هذه الجملة المستطيل أتوماتيكياً

ولن نناقش جملة DRAW بأى تفاصيل فى هذا الكتاب حيث أن قواعد تعريف شكل السلاسل معقدة نوعاً ما . وعلى القارىء أن يرجع إلى كتيب بيسك المرجعى المتاح للحاسب الدقيق الخاص به لمزيد من المعلومات عن هذا الموضوع .

١٢ ــ ٤ الرسوم المتحركة ANIMATIONS

إن إحدى التطبيقات على الحاسبات الدقيقة الأكثر تشويقاً وتسلية هى عروض الرسوم المتحركة . ومثل هذه التطبيقات تعطى أساساً للعديد من ألعاب الحاسبات وهى جزء مهم لكثير من البرامج التعليمية والفنية .

وقد قمنا فعلاً بعرض بعض الرسوم المتحركة البسيطة فى الأمثلة ١٢ ــ ٨ (نقط فى الفارغ) ، ١٢ ــ ١١ (الفن الحركى) ، ١٢ ــ ١٤ (المستطيلات المتمددة)، ١٢ ــ ١٨ (الدوائر المتمددة) و ١٢ ــ ٢٧ (المنطاد بنص متحرك) . وكل هذه الرسوم المتحركة

تعتمد على نفس الفكرة : عرض أى شيء أو سطر فى نص ، تكوين تأخير قصير للوقت ومسح الشيء . ثم التحرك إلى موضع قريب وإعادة السلسلة .

ويهتم هذا القسم بالرسوم المتحركة التى تتضمن حركة أشياء مليئة . وسنرى أولاً كيف يتم ذلك باستخدام بعض الجمل المعروفة الآن مثل CIRCLE و PAINT.وسنعرض طريقة بديلة تعتمد على جملتين جديدتين PUT,GET . وسنرى أن الطريقة الثانية تعطى بعض المزايا المتميزة عن الطريقة الأولى .

مثال ١٢ ــ ٢٨ محاكاة كرة مرتدة Simulation of a Bouncing Ball

دعنا الآن نعتبر رسم متحرك بسيط والذى يسمح لكرة بأن تتحرك بحرية داخل فراغ محدود . وينشأ الرسم المتحرك أولاً بعرض الكرة عند موضع معين وملئها بلون وتتوقف وقتاً قصيراً ثم تمسح الكرة . (أى يعاد عرض الكرة باستخدام لون الخلفية) . ثم تتحرك إلى موضع قريب وتتكرر الخطوات كلها . وعندما يقابنا حاجز (أى حائط) فإن الكرة تغير اتجاهها وعلى ذلك تظهر الكرة وكأنها ترتد عن الحائط .

ويبين شكل ١٢ ــ ٢٤ برنامجا كاملاً ميكروسوفت بيسك متقدم مكتوباً لحاسب IBM الشخصى . وهذا البرنامج يستخدم بيانيات التحليل العالى . وعلى ذلك فإنه لا يتضمن استخدام ألوان متعددة (يدعم حاسب IBM الشخصى أسود وأبيض فقط فى أسلوب التحليل العالى) . وفى هذا البرنامج يتم توليد موضع الكرة عند البداية عشوائياً وكذلك المسافة بين المواضع المتتالية .

```
10 '*********** BOUNCING BALL ***********
20 '
30 KEY OFF : CLS
40 DEFINT A-Z
50 RANDOMIZE : CLS
60 SCREEN 2
70 LINE (0,0)-(639,199),,B
80 LINE (10,5)-(629,194),,B
90 PAINT (5,2)
100 X=20+INT(600*RND) : Y=20+INT(160*RND)
110 DX=5*(INT(4*RND)+1) : DY=5*(INT(4*RND)+1)
120 '
130 '*** BEGIN LOOP ***
140 '
150 CIRCLE (X,Y),10,0
160 PAINT (X,Y),0
170 X1=X+DX : Y1=Y+DY
180 IF X1 < 21 THEN X1=21 : DX=-DX : GOTO 200
190 IF X1 > 618 THEN X1=618 : DX=-DX
200 IF Y1 < 10 THEN Y1=10 : DY=-DY : GOTO 220
210 IF Y1 > 189 THEN Y1=189 : DY=-DY
220 CIRCLE (X1,Y1),10
230 PAINT (X1,Y1)
240 X=X1 : Y=Y1
250 GOTO 150
260 END
```

شكل ١٢ ــ ٢٤

دعنا نعتبر هذا البرنامج بالتفصيل . يحذف سطر 30 أى عروض سابقة لتعريفات مفتاح الدالة ويمسح الشاشة . وسطر 40 يقوم بتعريف كل المتغيرات لتكون من نوع الرقم الصحيح . وسطر 50 يستهل مولد العدد العشوائى ثم يمسح الشاشة ، وسطر 60 يحدد بيانيات التحليل العالى . تولد السطور 70 إلى 90 مستطيلاً مصمتاً حول الحواف الخارجية للشاشة ، معطية بذلك العائق الذى يحتوى على حركة الكرة . ثم يتم توليد نقطة بداية عشوائية فى سطر 100 ، ويتم توليد مجموعة من القيم عشوائياً لزيادة كل حركة فى سطر 110 .

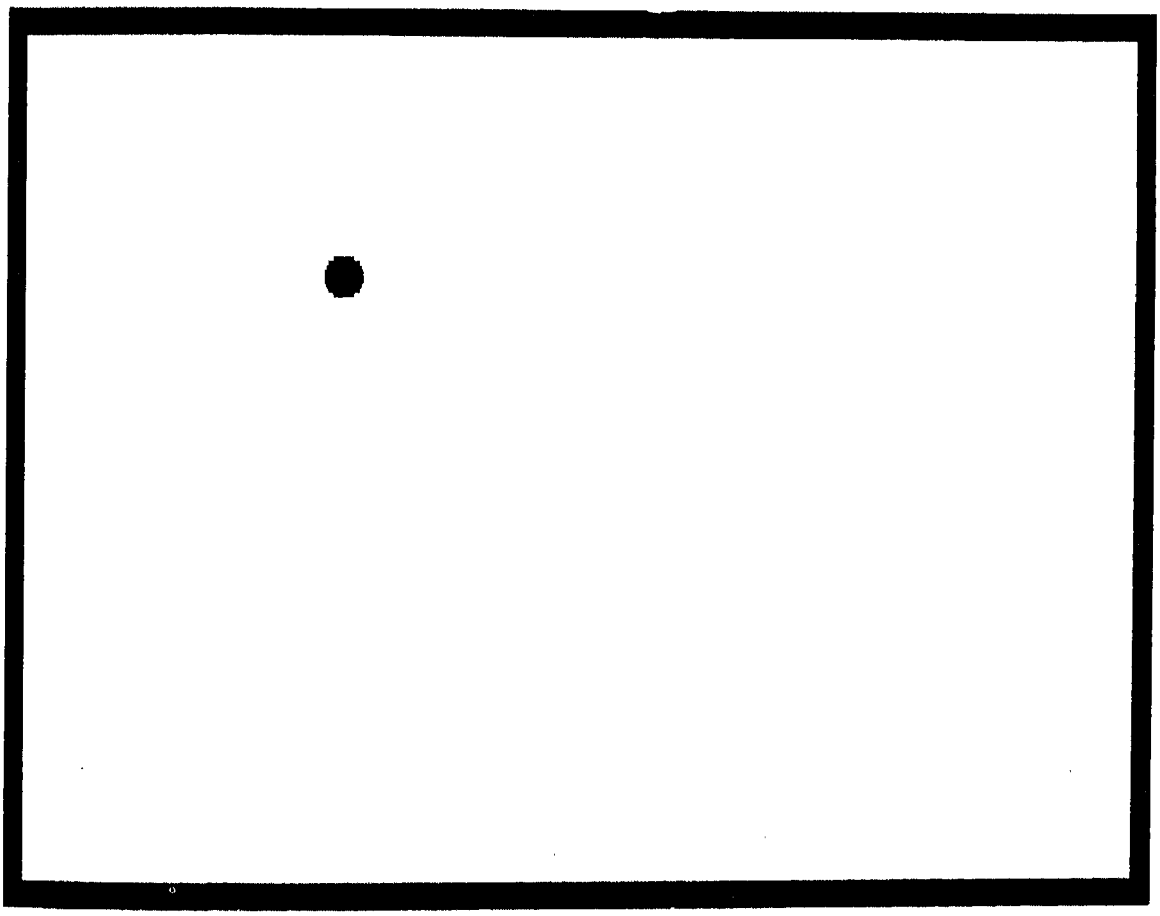
ويحتوى الجزء المتكرر من البرنامج على السطور 150 إلى 250 . ويسبب السطران 150 و 160 مسح الكرة من موضعها الحالى . (أى تعرض الكرة بلون أسود وهو لون الخلفية فى الموضع الحالى) . ثم يولد سطر 170 موضعاً جديداً . وفى السطور 180 إلى 210 يتم اختيار النقطة الجديدة

لتحديد ما إذا كانت الكرة ستذهب إلى ما بعد الحوائط . وإذا كان ذلك ، فإن اتجاه الحركة سيتغير وينطوى ذلك خداع بأن الكرة ارتدت عن الحوائط وتعود إلى الفراغ الداخلى المحدود .

يتسبب السطران 220 و 230 فى عرض الكرة فى الموضع الجديد ثم ملئها بلون أبيض ثم يتم تخفيض قيم الإحداثيات الجديدة للمتغير X و Y فى سطر 240 . ويستخدم هذان المتغيران لمسح الكرة أثناء المرور التالى خلال الحلقة التكرارية .

ويبين شكل ١٢ ــ ٢٥ نوع المخرجات التى تنشأ عند تنفيذ البرنامج . ومع كل ، فلابد للقارىء أن يتذكر أن الكرة تظهر وكأنها فى حركة ثابتة عن تنفيذ البرنامج الحقيقى .

ورغم أن طريقة الرسوم المتحركة السابقة تعمل فى حالة جيدة ، فإن الرسوم المتحركة تكون بطيئة نسبياً ويصاحبها أحياناً ارتعاش مضايق . ومن الممكن لبعض نسخ ميكروسوفت بيسك أن تولد رسوماً متحركة أسرع وخالية من الارتعاش عن طريق استخدام جملتى GET و PUT . وبهذه الطريقة يتم توليد رسوم متحركة للشيء مرة واحدة . وتقوم جملة GET بتحويل الشيء فى الشاشة إلى صف ثم تقوم جملة PUT بعد ذلك بالتحويل من الكتلة المتراصة ثانية إلى الشاشة عند موضع مختار . وإذا تم وضع الشيء مباشرة على نفسه ، فإنه يظهر بلون عكس اللون السابق (أى أسود يحل محل الأبيض) ، وعلى ذلك بتنفيذ زوج من PUT عند نفس الموضع فإنه يمكن عرض الشيء أتوماتيكياً ثم مسحه (لاحظ أنه عند استخدام جملتى GET و PUT بهذه الطريقة لا يحدث خلط بجملتى GET و PUT المستخدمة فيما يتعلق بملفات البيانات العشوائية كما هو مبين فى قسم ٩ ــ ٤ ، مثال (٩ ــ ٣١) .



شكل ١٢ ــ ٢٥

والطريقة العامة إذاً هى توليد الشىء الذى يتحرك حول الشاشة ووضعه فى كتلة متراصة باستخدام جملة GET . وهذا يتم مرة واحدة فقط ويدخل البرنامج فى حلقة تكرارية والتى يتم فيها مسح الشىء من مكانه القديم (عن طريق جملة PUT) ، ويحدد موضع جديد ويعاد عرض الشىء فى هذا الموضع الجديد (ثانية عن طريق PUT) والرسم المتحرك ينشأ عن طريق تكرار المرور خلال الحركة التكرارية .

وتتكون جملة GET من كلمة GET يعقبها زوجان من الإحداثيات . ولابد أن ينحصر كل زوج من الإحداثيات بين قوسين تفصلهما فاصلة . وتقوم هذه الإحداثيات بتعريف الأركان المتقابلة للمستطيل الذى يحتوى على الشىء . ويفصل الشرطة (علامة ناقص) زوجى الإحداثيات . ويتبع الزوج الثانى من الإحداثيات اسم الكتلة المتراصة الذى يحوى الشىء ، أى GET (X,Y)—(X+20,Y+20), FIGURE

تعتمد أبعاد الكتلة المتراصة على كل من حجم الشىء (حقيقة حجم المستطيل المحصور) ومستوى التحليل . وهناك صيغة رياضية لحساب الأبعاد المطلوبة التى تستخدم هذين العاملين . وعلى القارىء أن يرجع إلى الكتيب المرجعى لحاسبه الخاص لمعلومات عن هذا الموضوع .

وبمجرد تخزين الشىء فى الكتلة المتراصة المطلوبة يتم إعادة عرضه بواسطة جملة PUT . وتتكون هذه الجملة من كلمة PUT يعقبها زوج مفرد من الإحداثيات محصورة بين قوسين وتفصلهما فاصلة . وتمثل هذه الإحداثيات الركن الأعلى الأيسر فى المثلث الذى يحتاج للشىء . ويعقب الإحداثيات بعد ذلك اسم الكتلة المتراصة أى PUT (X,Y),FIGURE

مثال ١٢ ــ ٢٩ العودة إلى الكرة المرتدة **The Bouncing Ball Revisited**

دعنا نعتبر ثانياً الرسم المتحرك لكرة داخل سياج كما هو موضح فى مثال ١٢ ــ ٢٨ . ومع كل سنقوم بتوليد الرسم المتحرك باستخدام جمل GET و PUT .

يحتوى شكل ١٢ ــ ٢٦ على البرنامج الكامل مكتوب مرة أخرى بميكروسوفت بيسك المتقدم لحاسب IBM الشخصى . ويشبه هذا البرنامج ذلك الموضح فى شكل ١٢ ــ ٢٤ . ومع كل ، فهناك بعض الاختلافات الهامة . لاحظ إضافة جملة DIM فى سطر 70 ، والذى يقوم بتعريف BALL على إنها كتلة متراصة من 34 عنصر . لاحظ أيضاً أن المسافة بين موضعين متتابعين (تم توليدها فى سطر 120) تتناقض حتى تستوى حركة الكرة . ولم يكن هذا عملياً فى النسخ السابقة ، لأن الرسم المتحرك كان بطيئاً إلى حد ما .

```
10 '************ BOUNCING BALL ************
20 '
30 KEY OFF : CLS
40 DEFINT A-Z
50 RANDOMIZE : CLS
60 SCREEN 2
70 DIM BALL(34)
80 LINE (0,0)-(639,199),,B
90 LINE (10,5)-(629,194),,B
100 PAINT (5,2)
110 X=20+INT(600*RND) : Y=20+INT(160*RND)
120 DX=1+INT(10*RND) : DY=1+INT(10*RND)
130 CIRCLE (X,Y),10
140 PAINT (X,Y)
150 GET (X-10,Y-10)-(X+10,Y+10),BALL
160 '
170 '*** BEGIN LOOP ***
180 '
190 PUT (X-10,Y-10),BALL
200 X1=X+DX : Y1=Y+DY
210 IF X1 < 21 THEN X1=21 : DX=-DX : GOTO 230
220 IF X1 > 618 THEN X1=618 : DX=-DX
230 IF Y1 < 10 THEN Y1=10 : DY=-DY : GOTO 250
240 IF Y1 > 189 THEN Y1=189 : DY=-DY
250 PUT (X1-10,Y1-10),BALL
260 X=X1 : Y=Y1
270 FOR C=1 TO 5: NEXT C
280 GOTO 190
290 END
```

شكل ١٢ ــ ٢٦

تظهر الجملتان CIRCLE و PAINT مرة واحدة فقط فى البرنامج الحالى فى سطرى 130 و 140 على الترتيب . تعطى هاتان الجملتان تعريف شكل الكرة فى البداية . وتتسبب جملة GET فى سطر 150 بتخزين تعريف هذا الشكل فى كتلة متراصة تسمى BALL .

ويختلف جزء الحلقة التكرارية فى هذا البرنامج أيضاً بعض الشىء عن النسخة السابقة . ويتم استبدال أزواج جملتى CIRCLE و PAINT بجمل PUT فى سطرى 190 و 250 . ويمسح الأول منهما فى سطر 190 الكرة فى الموضع السابق . ويقوم الثانى بعرض الكرة فى موضعها الجديد . لاحظ فى النهاية أن الحلقة التكرارية الفارغة FOR-TO قد تمت إضافتها فى سطر 270 . وتهدف هذه الحلقة التكرارية إلى إبطاء الرسم المتحرك حتى يكون أكثر تشويقاً للعين .

وسوف تظهر الشاشة مرة أخرى عند تنفيذ البرنامج فى شكل ١٢ ــ ٢٥ إلا أن الحركة الآن ستكون أكثر سرعة وأكثر تسوية . ونهيب بالقارىء أن يقوم بالتنفيذ الفعلى لكل من البرنامجين حتى يكون أكثر تفهماً للاختلافات فى الرسوم المتحركة الناتجة .

ولا تكون الرسوم المتحركة التى تتكون بهذه الطريقة محدودة بشىء واحد متحرك إذ يمكن أيضاً تحريك عدة أشياء حول الشاشة باستخدام أزواج متعددة من جمل PUT . ولابد من تخزين كل شىء فى كتلة متراصة منفصلة . ويبين المثال التالى هذه الطريقة بالتفصيل .

مثال ١٢ ــ ٣٠ لعبة كرة التجديف A Game of Paddleball

هذه نسخة بسيطة من لعبة فيديو شائعة يشار إليها عادة بكرة التجديف . تنحصر الكرة بين ثلاث حوائط . وهناك مجداف متحرك صغير موضوع حيث كان يجب وضع الحائط الرابع عادة . ويمكن تحريك هذا المجداف إلى أعلى أو أسفل استجابة إلى وضع أحد وسائل التحكم مثل عصا التوجيه (سوف نفترض أن الحائط الغير موجود هو رأسى) . وإذا ما اصطدمت الكرة بالمجداف ، فإنها تعود ثانية إلى منطقة اللعب ، وإلا فإن الكرة تمر خلال الفتحة وتختفى .

والغرض من اللعبة هو التعرف مسبقاً على مسار الكرة حسبما تتحرك تجاه المنطقة المفتوحة ووضع المجداف بحيث يصدم الكرة ويعيدها ثانية إلى منطقة اللعب . وسيحصل اللاعب على نقطة لكل صدمة للكرة ويخسر نقطة واحدة إذا أخطأت الكرة المجداف . وسوف تعود الكرة فى الظهور تلقائياً عند موضع ما عشوائياً داخل منطقة اللعب بعد كل مرة تخطىء فيها الاصطدام مع المجداف .

يحتوى شكل ١٢ ــ ٢٧ على برنامج بيسك كامل لهذه اللعبة . والبرنامج مكتوب بميكروسوفت بيسك المتقدم لحاسب IBM الشخصى . ويحتوى على استخدام بيانيات التحليل المتوسط ، جمل تحكم عصا التوجيه وتحسينات الصوت لمصاحبة حركة الكرة . وعلى ذلك فإن البرنامج يحتوى على العديد من الخصائص الموضحة فى فصل ١٠ وكذلك طريقة الرسوم المتحركة .

دعنا الآن نعتبر الخصائص الرئيسية لهذا البرنامج . يحذف سطر 30 أى تعريفات سابقة لمفتاح الدالة ، ويمسح الشاشة ويعلن سطر 40 عن كل المتغيرات من نوع الرقم الصحيح . وتولد السطور 50 إلى 70 عرض نص مبدئى . ويستهل سطر 80 مولد العدد العشوائى . وسطر 90 يجعل الإشارة القادمة من زر 1 لعصا التوجيه مصاحباً للبرنامج الفرعى ، ثم يقوم تنشيط هذه المصاحبة (يستخدم زر 1 لإيقاف هذا البرنامج) . ويستدعى سطر 100 بيانيات التحليل المتوسط مع خلفية سوداء ولوحة ألوان رقم 1 (ألوان سماوى ، بنفسجى وأبيض) . ويتم إدخال الكتل المتراصة التى تحتوى على تعريفات شكل الكرة والمجداف فى سطر 110 ، ويتم تخصيص القيم الإبتدائية لبعض معالم البرنامج فى سطر 120 (لاحظ أن DX و DY تحددان السرعة الرأسية والأفقية للكرة كما هو موضح فى المثالين السابقين . وتحدد P موضع المجداف) .

وتولد السطور 160 إلى 180 الجدران الثلاثة ، ويسبب السطران 190 و 200 عرضاً مختصراً للنص عند أسفل الشاشة تحت منطقة العرض البيانى . ويحتوى النص على النتيجة وعلى تلقين يبين كيف تنتهى اللعبة .

وسطر 240 يتسبب فى توليد موضع الكرة عند البداية عشوائياً . ثم يتم توليد شكل الكرة وشكل المجداف ، ثم يتم تخزينهما فى الكتلتين المتراصتين المناظرتين فى السطرين 260,250 .

وتحتوى الحلقة التكرارية الرئيسية التى ينشأ فيها الرسم المتحرك على السطور 300 إلى 440،ويحدد سطر 300 المواضع الجديدة للكرة والمجداف . ويضبط السطران 310 و 320 موضع المجداف إذا ما كان أعلى ارتفاعاً أو أقل إنخفاضاً من اللازم . ويختبر السطران 330 و 340 موضع الكرة لمعرفة ما إذا كان يقع داخل مسار المجداف . وإذا كان كذلك فإنه يتم تداول برنامجين فرعيين مختلفين . ويتوقف الاختيار عما إذا كان المجداف يصدم الكرة أم لا حسب ما يحدده الاختبار فى سطر 340 .

```
10 '************ PADDLEBALL GAME ************
20 '
30 KEY OFF : CLS
40 DEFINT A-Z
50 LOCATE 4,1 : PRINT "Welcome to PADDLEBALL"
60 LOCATE 8,1 : PRINT "Rules: 1 point for each hit, "
70 LOCATE 10,7 : PRINT "-1 point for each miss"
80 LOCATE 14,1 : RANDOMIZE : CLS
90 ON STRIG(0) GOSUB 610 : STRIG(0) ON
100 SCREEN 1 : COLOR 0,1
110 DIM BALL(34),PADDLE(22)
120 DX=10 : DY=10 : P=8 : SCORE=0
130 '
140 '****** DRAW THE BORDER ******
150 '
160 LINE (0,0)-(319,5),1,BF
170 LINE (0,178)-(319,183),1,BF
180 LINE (314,6)-(319,177),1,BF
190 LOCATE 25,1 : PRINT "Score:";SCORE;
200 LOCATE 25,20 : PRINT "To stop, press B1";
210 '
220 '****** DRAW THE INITIAL FIGURES ******
230 '
240 X=10+INT(300*RND) : Y=10+INT(164*RND)
250 CIRCLE (X,Y),5 : PAINT (X,Y) : GET (X-5,Y-5)-(X+5,Y+5),BALL
260 LINE(0,P)-(6,P+20),3,BF : GET (0,P)-(6,P+20),PADDLE
270 '
280 '****** MAIN LOOP ******
290 '
300 X1=X+DX : Y1=Y+DY : DUMMY=STICK(0) : P1=199*STICK(1)/100
310 IF P1 < 6 THEN P1=6 : GOTO 330
320 IF P1 > 157 THEN P1=157
330 IF X1 >= 12 THEN 350
340 IF Y1 >= P1-2 AND Y1 <= P1+22 THEN GOSUB 480 ELSE GOSUB 530
350 IF X1 > 308 THEN X1=308 : DX=-DX : SOUND 1000,2
360 IF Y1 < 11 THEN Y1=11 : DY=-DY : SOUND 1000,2 : GOTO 380
370 IF Y1 > 172 THEN Y1=172 : DY=-DY : SOUND 1000,2
380 PUT (X-5,Y-5),BALL     'erase ball
390 PUT (0,P),PADDLE       'erase paddle
400 PUT (X1-5,Y1-5),BALL   'redraw ball
410 PUT (0,P1),PADDLE      'redraw paddle
420 X=X1 : Y=Y1 : P=P1
430 FOR C=1 TO 10: NEXT C
440 GOTO 300
450 '
460 '****** PADDLE HIT THE BALL ******
470 '
480 X1=12 : DX=-DX : SCORE=SCORE+1 : SOUND 1000,2
490 LOCATE 25,7 : PRINT SCORE;: RETURN 360
500 '
510 '****** PADDLE MISSED THE BALL ******
520 '
530 IF X1 >= 5 THEN RETURN 360
540 PUT (X-5,Y-5),BALL : PUT (0,P),PADDLE
550 DX=10 : DY=10 : SCORE=SCORE-1 : SOUND 50,2
560 LOCATE 25,7 : PRINT SCORE;
570 FOR I=1 TO 1000 : NEXT I : RETURN 240
580 '
590 '****** END THE GAME ******
600 '
610 END
```

شکل ۱۲ ــ ۲۷

وتختبر السطور 350 إلى 370 موضع الكرة بالنسبة إلى الجدران الثابتة الثلاثة . فإذا ما تداخل موضع الكرة المحسوب مع موضع الجدار فإنه يتم إجراء تعديل فى موضع الكرة كما يتم انعكاس سرعتها مكوناً بذلك خداعاً بأن الكرة ارتدت عن الحائط . ويصاحب الارتداد صوتاً ذا نغمة عالية .

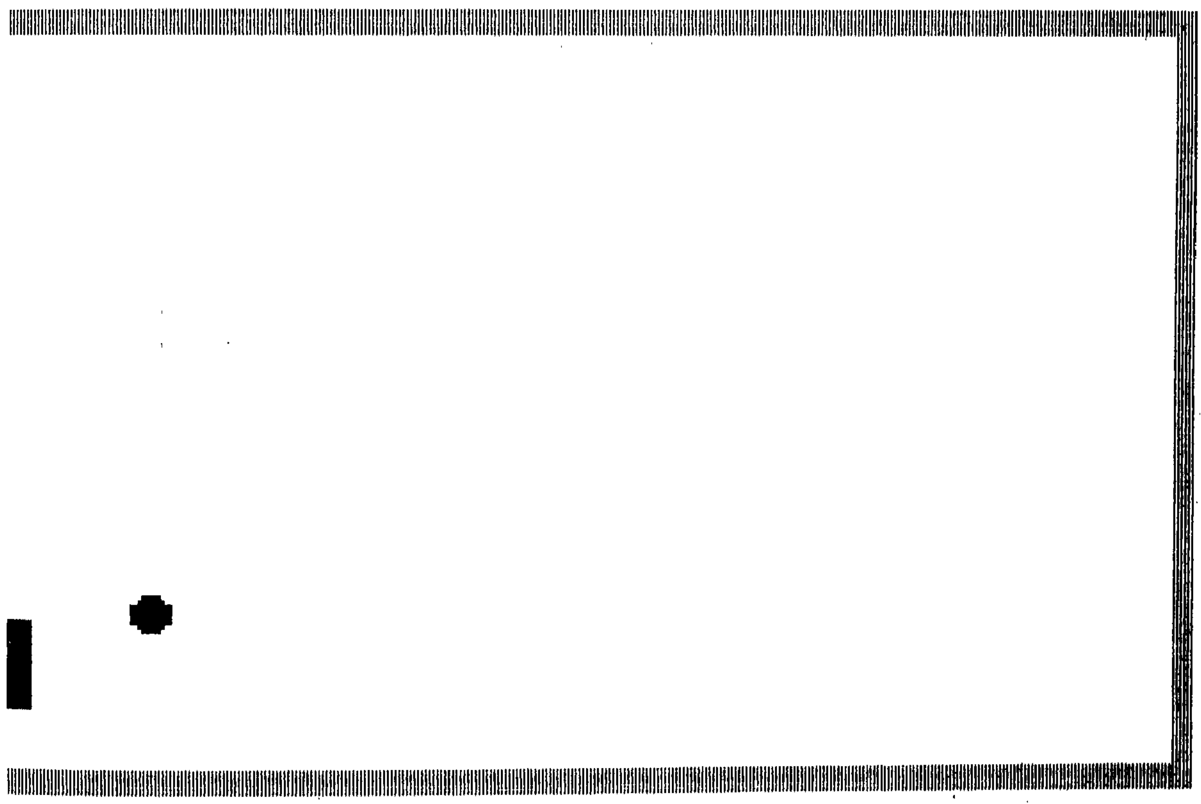
ويسبب السطران 380 و 390 مسح الأشياء من موضعها القديم . ويسبب السطران 400 و 410 إعادة رسمها فى موضعها الجديد . ويستهل سطر 420 قيم المواضع القديمة ، ويدخل فى سطر 430 تأخير وقت قصير وذلك تجنبا للرعشات الزائدة أساساً . ويعيد سطر 440 التحكم إلى سطر 300 ، وهكذا يبدأ مرور آخر خلال الحركة التكرارية .

ويقوم السطران 480 و 490 بعمل برنامج فرعى يتم تداوله عندما يضرب الجداف الكرة . وينشىء هذا البرنامج الفرعى أساساً تأثير ارتداد مشابه لما ينتج عندما تلمس الكرة أحد الجدران . وتزداد الأهداف المتراكمة مقدار 1 ، ويتم عرض القيم النهائية عند أسفل الشاشة .

وتقوم السطور 530 إلى 570 بعمل برنامج فرعى يمكن تداوله عندما يخطىء الجداف الكرة . ويعرض السطر 530 الكرة ببساطة بالقرب من الحافة اليسرى للشاشة إذا ما كان هذا هو موضعها المحسوب ، وبهذا يمنع الكرة من الاختفاء عن النظر قبل الأوان .

```
Welcome to PADDLEBALL
Rules: 1 point for each hit,
      -1 point for each miss
Random number seed (-32768 to 32767)? 12345
```

شكل ١٢ ـــ ٢٨



Score:-5   To stop, press B1

شكل ١٢ ـــ ٢٩

ويقوم بمحاكاة تأثير اختفاء الكرة عن النظر فى سطر 540 . ويقوم نفس السطر بمسح كل من الكرة والمجداف . ويسبب سطر 550 اعادة وضع السرعات إلى قيمتها الأولية الموجبة مسبباً بذلك وصول الكرة إلى أسفل الركن الأيمن للشاشة عندما تعود للظهور عند موضع جديد عشوائى .

وتتناقص الأهداف المتجمعة بمقدار 1 ويتم توليد صوت قصير ذى نغمة منخفضة (لاحظ أنه يصدر صوت ذو نغمة مرتفعة عندما ترتد الكرة عن سطح ولكننا نسمع صوتاً ذا نغمة منخفضة عندما يخطىء المجداف الكرة).ثم يتم عرض الهدف الجديد عند أسفل الشاشة (سطر 560) ، ويتم توليد تأخير وقت قصير (سطر 570) قبل عودة التحكم إلى سطر 240 مسبباً وضع كرة جديدة فى اللعب .

وفى النهاية فإن جملة END فى سطر 610 هى فى الحقيقة البرنامج الفرعى الذى يتم تداوله عندما يتم الضغط على زر 1 على عصا التوجيه . وهذا التصاحب هو الذى تم تنشيطه فى سطر 90 .

وشكل ١٢ ــ ٢٨ يبين عرض النص عند أول ظهور له عند تنفيذ البرنامج . وبمجرد وجود قيمة لمولد العدد العشوائى فإن الشاشة تتحول إلى بيانيات التحليل المتوسط مبينة جدران سماوية وكرة بيضاء ومجداف . وتظهر الكرة عند وضع عشوائى على الشاشة وتظل فى حركة ثابتة وترتد عن أى سطح تلمسه . ويبين شكل ١٢ ــ ٢٩ مظهر الشاشة كلما اقتربت الكرة من المجداف فى أعلى اليمين .

ومن خلال كل هذا فإن اللاعب يتحكم فى موضع المجداف وذلك بضبط عصا التوجيه . ويستمر تكرار حركة الإرتداد حتى يتم السماح للكرة بالاختفاء فى الجانب الأيسر للشاشة . ويرجع ذلك لتحديد موضع غير صحيح للمجداف ، ثم تعود الكرة للظهور عند موضع جديد بعد فترة قصيرة .

وتتزايد الأهداف المجمعة بمقدار 1 فى كل مرة يضرب المجداف الكرة ، وبالمثل فإن الأهداف المتجمعة تتناقص بمقدار 1 كلما أخطأ المجداف الكرة . ويتم دائماً عرض القيمة الحالية للأهداف المتجمعة فى أسفل الشاشة ، وتستمر اللعبة حتى يضغط اللاعب على زر 1 لعصا التوجيه .

ونهيب بالقارىء أن يقوم بتنفيذ هذا البرنامج إذا أمكن حتى يمكنه تفهم ماذا يحدث حقيقة . وهذا طبعاً مرغوب فيه جداً لكل البرامج من هذا النوع .

١٢ ــ ٥ بيانيات الحروف CHARACTER GRAPHICS

يمكن الحصول على تأثيرات بيانية معينة غالباً عن طريق معالجة ذكية لحروف النص . وتحتوى مثل هذه التأثيرات على عرض رسائل متحركة ، وتكوين أشكال ورسوم بسيطة . وقد رأينا فعلاً بعض الأمثلة من عروض الرسائل المتحركة بلف رأسى فى المثالين ١٠ ــ ١٠ و١٠ ــ ١٧ وتحرك أفقى كجزء من مثال ١٢ ــ ٢٧ .

ويبين برنامج ارتداد الكرة فى مثال ٦ ــ ٢٨ طريقة بسيطة ولكنها فعالة لاستخدام بيانيات الحروف على طابع السطور . ويسمح البرنامج الموضح فى مثال ١٠ ــ ١٣ باستخدام عصا توجيه لتكوين أشكال مختلفة كتجمعات من المنجمة (النجوم) . وكقاعدة فإن التأثيرات البيانية التى تنشأ بهذه الطريقة تكون أقل فعالية عن تلك التى تنشأ بوسيلة البيانيات . ومع كل فإن مثل تأثير البيانيات هذه يخدم غرضاً خصوصاً للحاسبات الدقيقة التى لا تدعم النص المنفصل ووسائل البيانيات .

ويوضح المثال التالى استخدام نوع بيانيات الحروف لتمثيل البيانيات الرقمية .

مثال ١٢ ــ ٣١ مولد خريطة الأعمدة A Barchart Generator

نفرض أن لدينا مجموعة من القيم العددية التى نريد عرضها فى شكل بيانى لتأكيد الطبيعة المنفصلة لكل قيمة . وغالباً تكون خريطة الأعمدة البيانية هى أفضل طريقة يتم بها ذلك والتى تمثل فيها كل قيمة بمستطيل يكون ارتفاعه متناسباً مباشرة مع قيمته المناظرة وعادة ما تعرض المستطيلات أفقياً وتكون كلها مرئية فى نفس الوقت .

وشكل ١٢ ــ ٣٠ يحتوى على برنامج ميكروسوفت بيسك مكتوب لحاسب IBM الشخصى والذى يتسبب فى توليد خريطة أعمدة بيانية لعدد يصل إلى ١٢ قيمة عددية . ومن المفروض أن تكون كل قيمة غير سالبة (أى أكبر من أو مساوية للصفر) . وسيتم عرض القيم أعلى كل مستطيل مناظر لها (أى أعلى العمود) وستكون لكل قيمة عنوان مناظر يتم عرضه تحت العمود . وستكون الأعمدة نفسها مكونة من تجمعات من النجوم .

```
10 '*************** BAR CHART GENERATOR **************
20 '
30 KEY OFF: CLS
40 WIDTH 80: SCREEN 0: COLOR 7,0
50 DIM LABEL$(12),Y(12)
60 LOCATE 1,20: PRINT STRING$(40,"*")
70 LOCATE 2,20: PRINT "*";SPC(38);"*"
80 LOCATE 3,20: PRINT "*";SPC(9);"BAR CHART GENERATOR";SPC(10);"*"
90 LOCATE 4,20: PRINT "*";SPC(38);"*"
100 LOCATE 5,20: PRINT STRING$(40,"*")
110 YMAX=0
120 '
130 '****** ENTER DATA AND FIND LARGEST Y ******
140 '
150 LOCATE 7,1: INPUT "Title: ",TITLE$
160 LOCATE 9,1: INPUT "How many data items? (1-12) ",ANS$: N=VAL(ANS$)
170 IF N < 1 OR N > 12 THEN BEEP: LOCATE 9,29: PRINT SPACE$(6): GOTO 160
180 FOR I=1 TO N
190    LOCATE I+10: PRINT "I =";I
200    LOCATE I+10,15: INPUT "Value: ",ANS$: Y(I)=VAL(ANS$)
210    IF LEFT$(ANS$,1)="0" THEN 230
220    IF Y(I) <= 0 THEN BEEP: LOCATE I+10,21: PRINT SPACE$(6): GOTO 200
230    LOCATE I+10,35: INPUT "Label: ",LABEL$(I)
240    IF Y(I) > YMAX THEN YMAX=Y(I)
250 NEXT I
260 '
270 '****** GENERATE AND DISPLAY THE BAR CHART ******
280 '
290 CLS: W=60\N
300 FOR I=1 TO N
310    R=20-18*Y(I)\YMAX: IF R=20 THEN 370
320    FOR ROW=R TO 20
330       FOR COL=(I-1)*W+11 TO I*W+8
340          LOCATE ROW,COL: PRINT "*"
350       NEXT COL
360    NEXT ROW
370 NEXT I
380 '
390 '****** LABEL THE BAR CHART ******
400 '
410 FOR I=1 TO N
420    R=19-18*Y(I)\YMAX
430    C=9 + (I-1)*W + (W-LEN(STR$(Y(I))))/2
440    LOCATE R,C: PRINT Y(I)
450    C=10 + (I-1)*W + (W-LEN(LABEL$(I)))/2
460    LOCATE 21,C: PRINT LABEL$(I)
470 NEXT I
480 C=10 + (N*W-LEN(TITLE$))/2
490 LOCATE 23,C: PRINT TITLE$;: LOCATE 23,1
500 END
```

شکل ۱۲ ـ ۳۰

Annual Sales Increases

Increase (%)	*Year*
5.2	1985
7.8	1986
8.2	1987
6.7	1988
10.6	1989
12.3	1990

دعنا نختبر البرنامج ببعض التفصيل . تمسح السطور 30 إلى 110 الشاشة . وتقوم بتعريف الكتل المتراصة LABEL$,Y وتوليد عنوان واستهلال YMAX . وتولد السطور 150 إلى 250 سلسلة من التلقينات التفاعلية للبيانات الداخلة .

وأول بند مطلوب هو عنوان خريطة الأعمدة البيانية (سطر 150)يعقبه طلب إدخال عدد بنود البيانات (السطران 160و 170) . لاحظ أن هذا الطلب يحتوى على تصيد خطأ (سطر 170) والذى يحتاج إلى استجابة المستخدم لتكون قيمته بين 1 و 12 . وتقوم السطور 180 إلى 250 بعمل حلقة تكرارية FOR-TO والتى تلقن المستخدم لكل من بنود البيانات وكل عنوان يصاحبها . وتحتوى الحلقة التكرارية على تصيد خطأ (السطران 210و 220) الذى يمنع بعض الحروف بخلاف الأعداد من إدخالها للقيمة المطلوبة . وأيضاً فإن أكبر قيمة مدخلة توجد فى سطر 210

ويتم توليد الأعمدة فى السطور 290 إلى 370 ويسمح سطر 290 للبيانات المدخلة ويحسب عرض كل عمود . (وكلما زاد عدد الأعمدة صغر عرض كل عمود) . وتقوم السطور 300 إلى 370 بعمل حلقة تكرارية ثلاثية تستخدم لتوليد الأعمدة وتتسبب الحلقة الخارجية فى توليد الأعمدة المتتابعة (لاحظ أن النصف الأعلى لكل عمود والمحسوب فى سطر 310 يحسب على أساس النسبة بين القيمة المعطاة والقيمة العظمى . وتولد الحلقة التكرارية الوسطى (السطور 320 إلى 360) الصفوف التى تكون كل عمود وتولد الحلقة التكرارية الداخلة (السطور 330 إلى 350) الأعمدة داخل كل صف .

وتقوم السطور 410 إلى 490 بعنونة خريطة الأعمدة . وتسبب الحلقة التكرارية FOR-TO (السطور 410إلى 470) فى عرض قيم كل بيان على السطر أعلى كل عمود وظهور العنوان المناظر تحت كل عمود . ويتسبب السطران 480و 490 فى عرض عنوان الخريطة فى الأعمدة الأسفل أفقياً وعند المركز .

ونفرض الآن برنامجا يتم تنفيذه باستخدام مجموعة البيانات المدخلة التالية : ويبين شكل ١٢ ــ ٣١ الديالوج الذى تم توليده بجزء البرنامج للبيانات المدخلة (واستجابة المستخدم تحتها خط) . وشكل ١٢ ــ ٣٢ يبين خريطة الأعمدة البيانية المناظرة .

```
******************************************
*                                        *
*          BAR CHART GENERATOR           *
*                                        *
******************************************

Title: Annual Sales Increases

How many data items? (1-12) 6

I = 1          Value: 5.2          Label: 1985
I = 2          Value: 7.8          Label: 1986
I = 3          Value: 8.2          Label: 1987
I = 4          Value: 6.7          Label: 1988
I = 5          Value: 10.6         Label: 1989
I = 6          Value: 12.3         Label: 1990
```

شكل ١٢ ــ ٣١

وتحتوى بعض الحاسبات الدقيقة على مجموعة خاصة من بيانيات الحروف بالإضافة إلى 128 حرف ASCII قياسياً المبينة فى (ملحق هـ) . فمثلا يدعم حاسب IBM الشخصى 256 حرف مختلف متضمناً 128 حرفا خاصا . والعديد من هذه هى حروف بيانية (رموز بيانية والحروف المستخدمة فى اللغات الأجنبية) . وقد استخدمنا فعلاً أحد بيانيات الحروف هذه لتكوين كتل من الضوء فى مثال ١٢ ــ ١٠ (انظر سطر 50 فى شكل ١٠ ــ ٧) . واستخدام بيانيات الحروف هذه يعطى إمكانية تحسين معنوى لعروض بيانيا .

```
                                                              12.3
                                                            *********
                                                            *********
                                                  10.6      *********
                                                *********   *********
                                                *********   *********
                           8.2                  *********   *********
               7.8      *********               *********   *********
            *********   *********      6.7      *********   *********
            *********   *********   *********   *********   *********
            *********   *********   *********   *********   *********
   5.2      *********   *********   *********   *********   *********
*********   *********   *********   *********   *********   *********
*********   *********   *********   *********   *********   *********
*********   *********   *********   *********   *********   *********
*********   *********   *********   *********   *********   *********
*********   *********   *********   *********   *********   *********
*********   *********   *********   *********   *********   *********
*********   *********   *********   *********   *********   *********
*********   *********   *********   *********   *********   *********
  1985        1986        1987        1988        1989        1990

                      Annual Sales Increases
```

شكل ١٢ ــ ٣٢

مثال ١٢ ــ ٣٢ مولد محسن بخريطة الأعمدة البيانية . An Improved Barchart Generator

يحتوى شكل ١٢ ــ ٣٢ على برنامج آخر لميكروسوفت بيسك مكتوب لحاسب IBM الشخصى لتوليد خريطة أعمدة بيانية كما هو مبين فى مثال ١٢ ــ ٣١ . هذا البرنامج يشبه البرنامج المعروض فى شكل ١٢ ــ ٣٠ . ونرى الآن إشارة إلى بيانيات الحروف الخاصة متاحة على حاسب IBM الشخصى . CHR$ (200) و CHR$ (186) و CHR$ (187) و CHR$ (205) و CHR $ (201) .

لاحظ مثلاً أن السطور 60 إلى 100 تحتوى على دالة تستدعى ضمن جمل PRINT المختلفة . واستدعاء هذه الدالة يتسبب فى وضع العنوان «BAR CHART GENERATOR» داخل برواز محاط بمستطيل مزدوج كما هو مبين فى شكل ١٢ ــ ٣٤ .

ويحدد سطر 110 أيضاً الحروف CHR$ (177) ، CHR$ (178) للمتغير BAR1 $ و BAR2 $ على الترتيب. ويتم طبع الحروف فى سطر 350 مسببة بذلك تكوين الأعمدة المستطيلة من كتل مستطيلة مظللة جزئياً بدلاً من النجوم . ويبين شكل ١٢ ــ ٣٥ هذا التأثير . لاحظ أن الأعمدة المتناوبة مبينة بتظليل مختلف .

ومن الأمور الشيقة مقارنة شكل ١٢ ــ ٣١ مع شكل ١٢ ــ ٣٤ . وواضح أن بيانيات الحروف الخاصة المستخدمة لتوليد شكل ١٢ ــ ٤٠ يحسن مظهر العرض . والأكثر إلفاتاً للنظر هو مقارنة شكل ١٢ ــ ٣٢ مع ١٢ ــ ٣٥ . (شكل ١٢ ــ ٣٥ يظهر أفضل على شاشة TV حيث كتلة الحروف المفردة داخل كل عمود سوف تلمس احدها الآخر مكونة مستطيلا مصمتًا) وعلى ذلك فإننا نرى أنه يمكن الحصول على عرض أفضل كثيراً فى الرؤية بإضافة جهد إضافى صغير من البرمجة .

```
10 '************** BAR CHART GENERATOR *************
20 '
30 KEY OFF: CLS
40 WIDTH 80: SCREEN 0: COLOR 7,0
50 DIM LABEL$(12),Y(12)
60 LOCATE 1,20: PRINT CHR$(201);STRING$(38,CHR$(205));CHR$(187)
70 LOCATE 2,20: PRINT CHR$(186);SPC(38);CHR$(186)
80 LOCATE 3,20: PRINT CHR$(186);SPC(9);"BAR CHART GENERATOR";SPC(10);CHR$(186)
90 LOCATE 4,20: PRINT CHR$(186);SPC(38);CHR$(186)
100 LOCATE 5,20: PRINT CHR$(200);STRING$(38,CHR$(205));CHR$(188)
110 BAR1$=CHR$(177): BAR2$=CHR$(178): YMAX=0
120 '
130 '****** ENTER DATA AND FIND LARGEST Y ******
140 '
150 LOCATE 7,1: INPUT "Title: ",TITLE$
160 LOCATE 9,1: INPUT "How many data items? (1-12) ",ANS$: N=VAL(ANS$)
170 IF N < 1 OR N > 12 THEN BEEP: LOCATE 9,32: PRINT SPACE$(6): GOTO 160
180 FOR I=1 TO N
190     LOCATE I+10: PRINT "I =";I
200     LOCATE I+10,15: INPUT "Value: ",ANS$: Y(I)=VAL(ANS$)
210     IF LEFT$(ANS$,1)="0" THEN 230
220     IF Y(I) <= 0 THEN BEEP: LOCATE I+10,21: PRINT SPACE$(6): GOTO 200
230     LOCATE I+10,35: INPUT "Label: ",LABEL$(I)
240     IF Y(I) > YMAX THEN YMAX=Y(I)
250 NEXT I
260 '
270 '****** GENERATE AND DISPLAY THE BAR CHART ******
280 '
290 CLS: W=60\N
300 FOR I=1 TO N
310     R=20-18*Y(I)\YMAX: IF R=20 THEN 380
320     FOR ROW=R TO 20
330         FOR COL=(I-1)*W+11 TO I*W+8
340             LOCATE ROW,COL
350             IF I MOD 2 = 0 THEN PRINT BAR1$ ELSE PRINT BAR2$
360         NEXT COL
370     NEXT ROW
380 NEXT I
390 '
400 '****** LABEL THE BAR CHART ******
410 '
420 FOR I=1 TO N
430     R=19-18*Y(I)\YMAX
440     C=9 + (I-1)*W + (W-LEN(STR$(Y(I))))/2
450     LOCATE R,C: PRINT Y(I)
460     C=10 + (I-1)*W + (W-LEN(LABEL$(I)))/2
470     LOCATE 21,C: PRINT LABEL$(I)
480 NEXT I
490 C=10 + (N*W-LEN(TITLE$))/2
500 LOCATE 23,C: PRINT TITLE$;: LOCATE 23,1
510 END
```

شکل ۱۲ ــ ۳۳

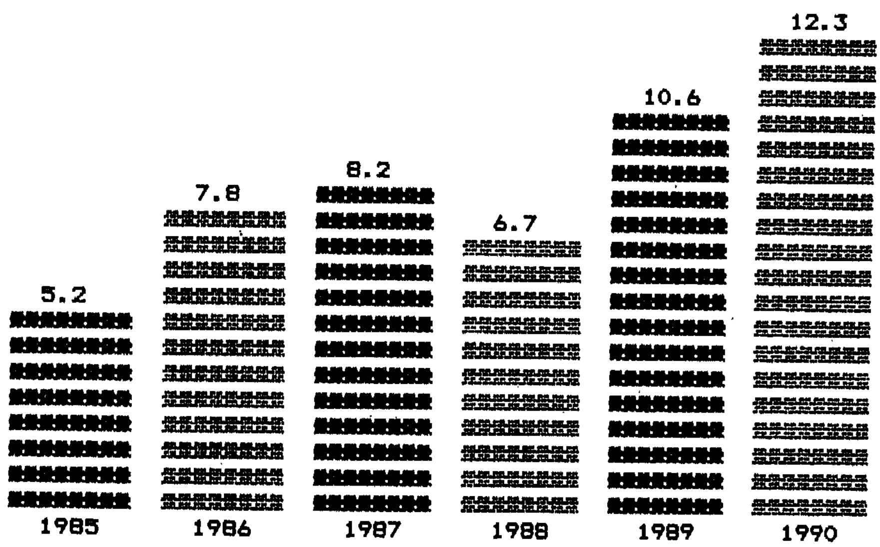
```
BAR CHART GENERATOR

Title: Annual Sales Increases

How many data items? (1-12) 6

I = 1        Value: 5.2       Label: 1985
I = 2        Value: 7.8       Label: 1986
I = 3        Value: 8.2       Label: 1987
I = 4        Value: 6.7       Label: 1988
I = 5        Value: 10.6      Label: 1989
I = 6        Value: 12.3      Label: 1990
```

شكل ١٢ ــ ٣٤



Annual Sales Increases

شكل ١٢ ــ ٣٥

اسئلة للمراجعة

Review Questions

١٢ - ١ حدد أن كانت أولا نسخة البيسك المتاحه لحاسبك الدقيق الخاص تحوى جمل بيانات خاصة . واذا كانت كذلك هل هى نفسها كما تم شرحها فى هذا الفصل ؟

١٢ - ٢ ما هو مستوى التحليل المعد لخواص البيانيات فى حاسبك الدقيق الخاص ؟ هل هناك انماط بيانات عديدة متاحة ؟

١٢ -٣ هل اللون متاح على حاسبك الدقيق الخاص ؟ اذا كان كذلك فكم عدد الألوان المختلفة المتاحة فى نمط النصوص ؟ وما هو عدد المتاح فى نمط البيانيات ؟

١٢ - ٤ ما هى النقطة ؟ وفيما تستخدم ؟

١٢ - ٥ ما الغرض من جملة SCREEN ؟ لخص القواعد التى تطبق لاستخدامها .

١٢ - ٦ ما الغرض من جملة PSET ؟ لخص القواعد التى تطبق لاستخدامها .

١٢ - ٧ ما الغرض من جملة PRESET ؟ وكيف تختلف عن جملة PSET ؟ لخص القواعد التى تطبق لاستخدامها .

١٢ - ٨ كيف يمكن توليد نقط مفرده فى نسختك من البيسك ؟

١٢ - ٩ ما الغرض من جملة LINE ؟ لخص القواعد التى تطبق لاستخدامها .

١٢ - ١٠ كيف يمكن توليد خط مفرد فى نسختك من البيسك ؟

١٢ - ١١ اشرح كيف ان عرضا على الشاشة يتكون كلياً من نقط فردية أو خطوط (ليست أشكال) يمكن أن ينشط ، أى كيف يمكن أن تعرض نقط أو خطوط يمكن أن تقرض وفيما بعد تمسح مثلما فى برنامج (الفن الحركى) .

١٢ - ١٢ كيف يمكن للجملة LINE أن تستخدم لتوليد مستطيل مغلق ؟ كيف يمكن لهذا المستطيل أن يُملأ بلون معين ؟

١٢ -١٣ ما الغرض من جملة ؟ لخص القواعد التى تطبق لاستخدامها ؟.

١٢ - ١٤ ما الغرض من جملة PAINT ؟ لخص القواعد التى تطبق لاستخدامها .

١٢ - ١٥ كيف يمكن للجملتى CIRCLE و PAINT أن تستخدما لتوليد شكل دائرى يملأ بلون معين ؟

١٢ - ١٦ هل يمكن توليد دوائر مغلقة فى نسختك من البيسك ؟ اذا كان كذلك فكيف يتم ؟

١٢ - ١٧ كيف يمكن للجملة CIRCLE أن تستخدم لتوليد أقواس بدلاً الدوائر الكاملة ؟ كيف يمكن لهذه الاقواس أن تتصل بنقطة الأصل لتكون أشكال دائرية ؟

١٢ - ١٨ كيف يمكن للجملة CIRCLE أن تستخدم لتوليد قطاعات ناقصة ؟ كيف يمكن تحديد حجم واتجاه (أى أفقى أو رأسى) لقطع ناقص ؟

١٢ - ١٩ هل يمكن توليد قطاعات ناقصة مغلقة فى نسختك من البيسك ؟ اذا كان كذلك كيف يمكن أن يتم ؟

١٢ - ٢٠ اشرح كيف ينشط نص معروض على الشاشة فى الأتجاه الأفقى داخل فراغ ثابت محدد ؟ .

١٢ -٢١ كيف يمكن تحريك شكل بسيط فى نسختك من البيسك ؟

١٢ -٢٢ ما الغرض من جملة GET عندما تستخدم فى وسيلة البيانيات ؟ لخص القواعد التى تطبق لاستخدامها

١٢ - ٢٣ ما الغرض من جملة PYT عندما تستخدم فى وسيلة البيانيات ؟ لخص القواعد التى تطبق لاستخدامها .

١٢ -٢٤ عند تكوين رسوم متحركة ، ما المميزات فى استخدام جملتى GET و PUT أكثر من ببساطة توليد لشكل بالوان تبادلية (مختلفة) (أولاً بلون الواجهة ثم بلون الخلفية ؟

١٢ - ٢٥ ما المقصود ببيانات الحروف ؟ كيف يمكن مقارنة نوعيات بيانات الحروف مع أنواع أخرى ؟

١٢ - ٢٦ هل تحتوى نسختك من البيسك على بيانات حروف خاصة ؟ إذا كانت كذلك فما هى ؟ وماذا يكون كود ASCII لها ؟

مسائل تكميلية

Supplementary Problems

المسائل التالية تختص بجمع المعلومات بدلاً من الحل الحقيقى للمسائل . اجب عن الأسئلة باستخدام النسخة الخاصة بك من بيسك الحاسب الدقيق .

١٢ - ٢٧ هل حاسبك الدقيق يحتوى على امكاية عروض البيانيات ؟ اذا كان كذلك ، هل يمكن لعرض بيانيات أن يتم تكوينها داخل برنامج البيسك ؟ كم عدد أنماط البيانيات المتاحة ؟ ما مستوى التحليل المتاح مع كل أسلوب ؟

١٢ - ٢٨ هل اللون متاح ؟ اذا كان كذلك ما هو عدد الالوان التى يمكن عرضها فى أى وقت واحد داخل اسلوب البيانيات ؟ ما هى هذه الألوان ؟ هل هناك تمييز بين الوان الواجهه والوان الخلفية ؟

١٢ -٢٩ حدد كيف يمكن تنفيذ الخصائص التالية فى البيسك ؟

(أ) كيف يمكن التوصل إلى كل اسلوب فى البيانيات ؟

(ب) كيف يمكن تحديد لون مفرد ؟

(جـ) كيف يمكن توليد نقط مفردة ؟ وكيف يمكن مسحها ؟

(د) كيف يمكن توليد خطوط ؟

(هـ) هل يمكن توليد مستطيلات بجملة واحدة ؟ واذا كان كذلك فهل يمكن ملء مستطيلات بلون مصمت ؟

(و) هل يمكن توليد دوائر بجملة واحدة ، واذا كان كذلك فهل يمكن ملء الدوائر بلون مصمت ؟

(ز) هل يمكن توليد قطاعات ناقصة بجملة مفردة ؟ واذا كان كذلك فهل يمكن ملء هذه القطاعات بلون مصمت ؟

(حـ) هل يمكن توليد اقواس دائرية أو قطاعات ناقصة بجملة واحدة ؟ .

(ط) هل هناك جمل خاصة متاحة لعمل رسوم متحركة ؟ واذا كان كذلك فما هى ؟ وكيف تعمل ؟

(ى) هل يمكن توليد نص من خلال اسلوب البيانيات ؟

(ك) هل هناك خصائص خاصه لبيانيات لم يتم شرحها فى هذا الفصل ؟ فاذا كانت كذلك ، فما هى ؟ وكيف تعمل ؟

(ل) هل يحتوى حاسبك الدقيق على حروف بيانية خاصة ؟ اذا كان كذلك ، فما هى وكيف يمكن التوصل لهذه الحروف فى البيسك ؟

مسائل للبرمجة

Programming Problems

١٢ – ٣٠ عدل البرامج الآتية بحيث يمكن تغيير لوحة الألوان ولون الخلفية فى أى وقت بضغط مفاتيح دالة مناسبة (مثل F1 لتغيير لوحة الألوان و F2 لتغيير لون الخلفية) .
(أ) نقط الفراغ (مثال ١٢ – ٨) .
(ب) الخطوط المتحركة (الفن الحركى) (مثال ١٢ – ١١) .
(جـ) المشكال (مثال ١٢ – ١٦)
(د) الدوائر المتمددة (مثال ١٢ – ١٨)

١٢ – ٣١ غير بعض القيم العددية التى يحتويها برنامج الفن الحركى فى مثال ١٢ – ١١ . جرب مع قيم مختلفة حتى تجد فئة من القيم والتى ترغب فيها تماما .

١٢ – ٣٢ عدل البرنامج فى مثال ١٢ – ١٦ (المشكال) بحيث يملأ الشاشة بعدد أكبر من المستطيلات . تأكد من اختزال حجم المستطيلات للتعويض عن الأعداد الأكبر عن العدد الكبير .

١٢ – ٣٣ عدل البرنامج فى مثال ١٢ – ١٦ (المشكال) بحيث تنتشر الاحداثيات المولدة عشوائيا بإنتظام على الحرف كله بدلا من أن تكون متجمعه بالقرب من مركز الشاشه . ماهو التأثير الذى تفضله ؟

١٢ – ٣٤ عدل البرنامج فى مثال ١٢ – ١٢ (الانحدار الخطى مع عرض بيانى) بحيث أن محورى X و Y يوضع لها عناوين ضمن البرنامج مما يسمح لتوليد عنوان بالقرب من أعلى الشكل البيانى .

١٢ – ٣٥ تم توسيع البرنامج فى مثال ١٢ – ١٢ بحيث يمكن توفيق معادلات أسيه وكثيرات الحدود لمجموعة من البيانات المدخلة ثم رسمها ، إستخدام نقط قريبه من بعضها لتمثيل المنحنيات عند رسم الأشكال .
(مثال٧ – ٢٢ يعطى المعادلات المناسبة لتوفيق كل نوع من المنحنيات على اساس طريقة المربعات الصغرى) . ضمن قائمه والتى تسمح للمستخدم أن يختار نوع المنحنى المطلوب . تأكد من تضمين ملقنات مناسبة واختبارات خطأ لكل من البيانات العددية واختبارات القائمة .

١٢ – ٣٦ عدل مولد الخريطه الدائرية فى مثال ١٢ – ٢٤ بحيث يملأ كل قطعه دائرية بلون غير لون الخلفية . تأكد من أن القطع المتجاورة ليس لها نفس اللون .

١٢ – ٣٧ عدل مولد خريطه الأعمدة فى مثال ١٢ – ٣٢ بحيث يمكن عرض مجموعة واحدة من البيانات المختلفة أو مجموعتين أو ثلاث فى نفس الوقت بشرط أن كل مجموعات البيانات يكون لها نفس العدد من الأعمدة . ضع الأعمدة المتناظرة واحدة بعد الأخرى بدون فراغات (أى ضع أول عمود لكل مجموعه بيانات فى تجميعه واحده ، ثم العمود الثانى لكل مجموعة بيانات .. الخ . إستخدام لون مختلف أو نموذج مختلف لكل مجموعة بيانات .

١٢ – ٣٨ أكتب برنامج بيسك للحاسب الدقيق الذى سوف يسمح لمجموعة من البيانات أن تدخل إلى الحاسب وعندئذ سوف يولد إما خريطة دائرية أو خريطة أعمدة (أنظر مثالى ١٢ – ٢٤ و ١٢ – ٣٢) ضمن قائمه والتى تسمح للمستخدم أن يختار النوع المطلوب من الأشكال . تأكد من تضمين تلقينات مناسبة واختبارت خطأ لكل من البيانات العددية واختيارات القائمة .

١٢ – ١٩ عدل البرنامج فى مثال ١٢ – ٢٧ « بالون مع نص متحرك» بحيث أن المستخدم يمكن أن يحدد الرسالة عبر منطقة العرض . ضمن تلقين للعبارة المطلوبة عند بداية البرنامج قبل رسم البالون .

١٢ – ٤٠ غير بعض القيم العددية الموجودة فى برنامج الكرة المرتدة فى مثال ١٢ – ٢٩ . جرب باستخدام قيم مختلفة . حتى تجد مجموعة من القيم التى ترغبها جيدا . قارن عمل البرنامج مع الذى حصلت عليه بإستخدام القيم الأصلية .

١٢ – ٤١ عدل لعبة الكره والمضرب المعطى فى مثال ١٢ – ٣٠ بحيث يمكن للمستخدم أن يعين درجه الصعوبة قبل بداية اللعبة . وحتى تكون اللعبة أصعب اجعل المضرب أصغر ودع الكرة تتحرك أسرع . ضمن قائمة تلقينات مناسبة واختبارات الخطأ لتساعد المستخدم فى اختيار الدرجة المطلوبه من الصعوبة .

١٢ – ٤٢ بتوسيع البرنامج المعطى فى مثال ١٠ – ١٣ (برمجه عصا التوجيه لتوليد بيانات الحروف) بحيث يمكن استخدام بيانات مختلفة للحروف . ضمن قائمة تسمح للمستخدم أن يختار الحروف البيانية . اسمح أيضا للمستخدم أن يغير لون الخلفية أو لون كل حرف جديد بالضغط على مفتاح داله مناسب . ضمن اختيار مفتاح الدالة فى القائمة . (اقتراح : احجز اقصى ٢٠ عمود يمين من الشاشه للقائمة . اسمح للباقى من الشاشه أن يستخدم كمنطقة البيانيات) .

١٢ – ٤٣ حل مسألة ١٢ – ٤٢ بإستخدام متجول بدلا من عصا التوجيه .

١٢ – ٤٤ اكتب برنامج بيسك للحاسب الدقيق ، مشابه للموجود فى مثال ١٠ – ١٣ والذى يسمح لعصا توجيه أو متجول أن بتوليد نقط فردية فى اسلوب البيانيات . ضمن البرنامج مايسمح لتغيير لون الخلفية أو لون كل نقطه جديده بضغط مفتاح دالة مناسب . عقد قائمة صغيرة عند قاع الشاشه لشرح استخدام مفاتيح الدوال .

١٢ – ٤٥ قم بتوسيع البرنامج فى مسألة ١٢ – ٤٤ بحيث يمكن رسم خطوط ومستطيلات ودوائر بالإضافة للنقط الفرديه . ضمن البرنامج املأ كل شكل مصمت بلون والذى يمكن للمستخدم أن يختار من القائمه .

أيضاً اسمح للمستخدم أن يغير لون الخلفية أو لون كل شكل جديد بضغط مفتاح دالة مناسب .

١٢ – ٤٦ اكتب برنامج بيسك للحاسب الدقيق والذى سوف يولد عرض شاشه كاملة لشعار مدرستك أو رمز الشركه . ضمن لون إذا كان ذلك متاح .

١٢ – ٤٧ اكتب برنامج بيسك للحاسب الدقيق والذى سوف يولد عرض شاشه كاملة لعلم كلا من الدوال الآتية (بقائمة لكى تزيد الصعوبة) .

(أ) اليابان	(هـ) الولايات المتحدة الأمريكية
(ب) فرنسا	(و) المملكة المتحدة
(جـ) الدنمارك	(ى) كندا
(د) النرويج	(ز) المملكة العربية السعودية

١٢– ٤٨ اكتب برنامج بيسك للحاسب الدقيق والذى سوف يولد محور X ومحور Y مقسماً بذلك الشاشه إلى اربع ارباع متساوية . عندئذ إعرض شكل بيانى للمعادلة

$$y = c_1 + c_2x + c_3x^2 + c_4x^3 + c_5x^4$$

مستخدما قيم من اختيارك الخاص لـ C_1 ، C_2 ، C_3 ، C_4 ، C_5 (لاحظ انه من الممكن أن نرسم خطوط مستقيمة ومعادلات درجة ثانية وثالثة .. إلخ . بوضع عدد معين من هذه الثوابت وبالصفر) .

اكتب البرنامج بطريقة معينة بحيث يمكن أن ينفذ بالتكرار ، بقيم مختلفة لـ C_1، C_2، C_3 و C_5 التى يمكن ادخالها من لوحة المفاتيح عند بداية كل تنفيذ .

١٢ – ٤٩ اكتب برنامج بيسك للحاسب الدقيق والذى سوف يولد عرض بيانى للمعادلة

$$y = 2e^{-0.1x} \sin (0.5x + c)$$

لقيم X تتغير من 0 إلى 60 ، و C = 0 ضمن محورى X و Y فى العرض البيانى . ضع عنوان لكل محور .

١٢ — ٥٠ قم بتوسيع مسألة ١٢ — ٤٩ بحيث يحدث توليد العرض البيانى داخل الحلقة التكرارية مع قيم (متزايدة) مختلفة مخصصة لـ ك أثناء كل مسار .

١٢ ـ ٥١ هل يمكن توليد عدد من القروض البيانية المشوقة بالمعادلة التى تمثل حلزون أرشميدس وهى :

$$x = ar \cos r$$
$$y = br \sin r$$

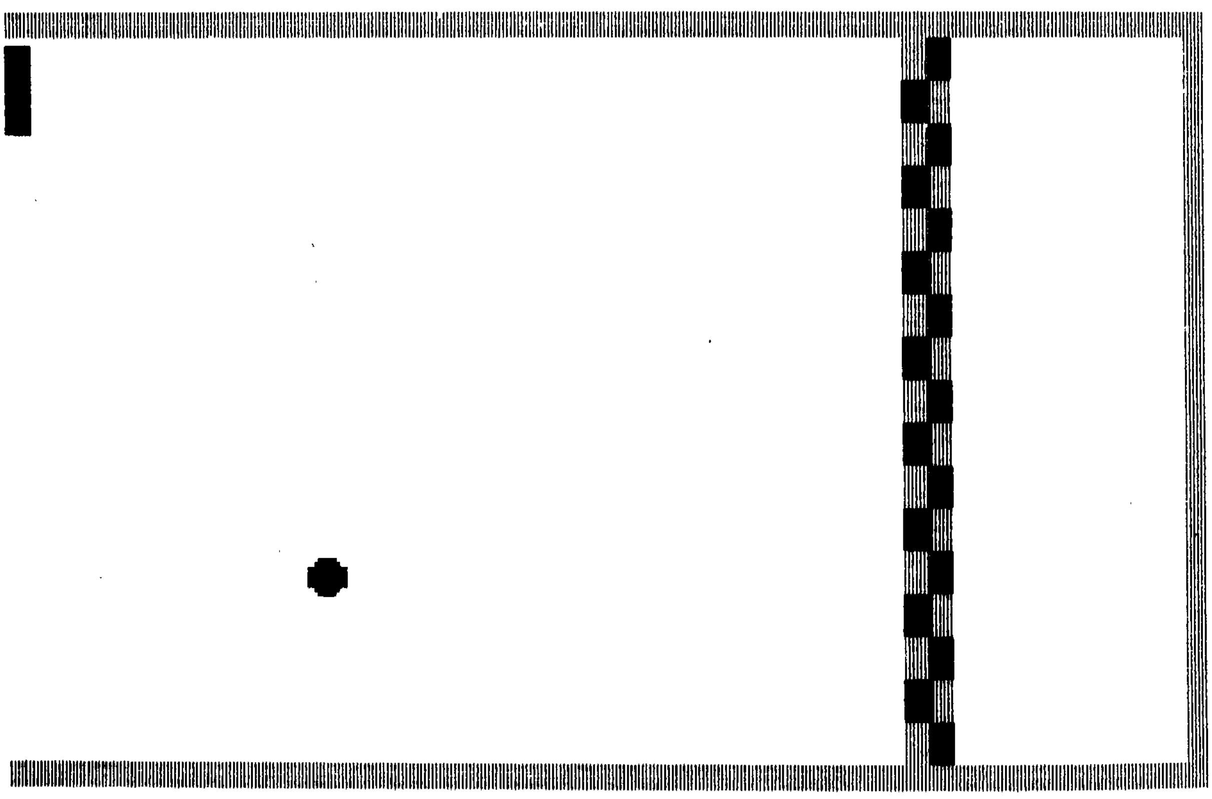
حيث a ، b ثوابت موجبة و ر تمثل زاوية O بالتقدير الدائرى .
اكتب برنامج بيسك للحاسب الدقيق والذى سوف يولد عرض بيانى لحلزون أرشميدس . اجعل الثوابت a ، b معالم مدخلة ولد سلسلة من القيم لـ ر (وهكذا سلسلة من قيم y,x) بجعل هاتين المعادلتين فى حلقة تكرارية FOR والتى تحتوى على معلمه STEP . ادخل قيمة معلمه STEP من لوحة المفاتيح ، برفقة قيمة a و b و عند بداية كل تنفيذ . (لاحظ أن كثير من العروض البيانية المختلفة يمكن توليدها بتعيين قيم مختلفة لـ a و b ومعلمه STEP .

١٢ ـ ٥٢ قم بتوسيع لعبة السرطان فى مثال ٦ ـ ٢٠ ليحتوى عرض بيانى للزهر بعد كل رمية . (لاحظ أن هذه المسألة يمكن أن تحل باستخدام أما بيانات من نوع النقط أوبيانيات حروف) .

١٢ ـ ٥٣ اكتب برنامج بيسك للحاسب الدقيق والذى سوف يسمح لشخص أن يلعب لعبة من تيك ـ تاك ـ تو أمام الكمبيوتر ضمن عرض يبين كلا من حركات اللاعبين مع محور X المألوف والصفرى ـ (انظر مسألة ٦ ـ ١٥٢ ز)

١٢ ـ ٥٤ قم بتوسيع برنامج الروليت المشروح فى مسألة ٦ ـ ٥٢ (ط) ليحتوى على عرض بيانى لعجلة الروليت (مع اللون إذا كان ممكن) . بين متى تأتى الكرة (الزجاجية) لتقف بعد كل لفه .

١٢ ـ ٥٥ قم بتوسيع برنامج البيجو المشروح فى مسألة ٦ ـ ٥٢ (ك) ليحتوى على عرض بيانى لكارت بنجو الرئيسى . (هذا الكارت سوف يحتوى على ٧٥ توفيقه ممكنه لعدد الحروف الممكنة فى خمس أعمدة ، معنون B11,N,G.O على الترتيب . أول عمود سوف يحوى الأرقام 1—15 ، والعمود الثانى سوف يحوى من 16—30 ... الخ) استخدم بيانات الحروف لتوليد العرض . عرف كل توقيت لرقم الحرف كما سحبها (أى ظلل الموضع على الكارت أو غير اللون) .



شكل ١٢ ـ ٣٦

١٢ ــ ٥٦ واحدة من أقدم العاب الفيديو والشعبية لعبة تسمى Brickout هذه اللعبة مشابهة للعبة مضرب التجديف المشروحة فى مثال ١٢ ــ ٣٠ فيما عدا إنه يوجد « حائط مستطيل من المظل » . موضوعه بالقرب من الجانب الأيمن لمنطقة اللعب ، كما تم مناقشته فى شكل ١٢ ــ ٣٦ .

إذا ضربت الكره طوبه ، ترتد الكره كما هو معتاد ولكن الطوبه تختفى واللاعب يحصل على نقطة . إذا مرت الكرة خلال قناة تنشأ عن الطوب المفقود بلمس حائط مستطيل ، عندئذ الكرة يمكن أن تستمر لترتد خلال الجزء الأيمن لمنطقة اللعب تضغط وتنزع الكتلة حتى تمر مرة ثانية من خلال قناة للجزء الأيسر من منطقة اللعب . اللعب يستمر حتى يختفى الطوب أو يفقد اللاعب الكره والمجداف . يسمح للاعب له بخمس كرات أثناء كل لعبة .

اكتب برنامج بيسك للحاسب الدقيق ليلعب لعبات متتالية من Brickout ، اجعل كل لعبة تالية أكثر صعوبة بجعل المضرب أصغر وجعل الكرة تتحرك بسرعة أكبر ابدأ درجة جديدة لكل لعبة ولكن حافظ على تسجيل لأعلى درجة .

إجابات لمسائل تكميلية مختارة .

١ – ٢٨ (أ) احسب مساحة مثلث معروف القاعدة والارتفاع .

(ب) احسب محيط مستطيل معروف الطول والعرض .

(حـ) احسب قيمة :

$$w = u + v$$
$$x = u - v$$
$$y = uv$$
$$z = u/v$$

حيث u , v معروفة القيم

(د) احسب قيمة :

$$y = 1 + x + \frac{x^2}{2} + \frac{x^3}{6}$$

حيث x معروفة القيمة

١ – ٢٩ (أ)

```
10 INPUT R
20 LET C=2*3.141593*R
30 PRINT R,C
40 END
```

يمكن كتابة الجملة 20 كالتالى :

```
20 LET C=6.283186*R
```

(ب)

```
10 INPUT B,H
20 LET L=(B↑2+L↑2)↑.5
30 PRINT B,H,L
40 END
```

(حـ)

```
10 INPUT U,V
20 LET W=(U−V)/(U+V)
30 PRINT U,V,W
40 END
```

(د)

```
10 INPUT X
20 LET Y=100*(1+X+2*X↑2+3*X↑3)
30 PRINT X,Y
40 END
```

١ – ٣٠ توجد الأخطاء التالية :

١ – يحتوى السطر الثانى على جملتين (يسمح بجملة واحدة فقط فى السطر) .

٢ – السطر الثالث لايحتوى على الكلمة الدالة LET (بعض نسخ البيسك تسمح بذلك) .

٣ – أرقام الجمل المتعاقبة فى البرنامج لاتتزايد (يجب أن يكون رقم السطر الرابع أكبر من 35 ولكن أقل من 40) .

٢ – ٤٣ (أ) 5
(ب) 8000 or 8E+3
(ﺟ) −1.8033E−9
(د) 0.33333333
(ﻫ) −7328500 or −7.3285E+6
(و) 2851 or 0.2851E+4
(ز) 0.2851E+10 or 2851E+6, etc.
(ح) −16752.47 or −1.675247E+4

٢ – ٤٤ (أ) صحيحة
(ب) صحيحة
(ﺟ) صحيحة
(د) لايمكن أن يحتوى الأس على رقم عشرى .
(ﻫ) صحيحة
(و) لاتسمح بالفواصل (,) .
(ز) الأس له قيمة كبيرة .
(ح) صحيحة
(ط) أرقام معنوية كثيرة جداً .
(ى) صحيحة
(ك) يجب أن يتبع الحرف E أس رقمى
(ل) الأس مكتوب بطريقة غير صحيحة (يجب أن يقرأ E-2)

٢ – ٤٥ (أ) صحيحة
(ب) صحيحة
(ﺟ) طويلة جداً لبعض نسخ البيسك .
(د) صحيحة
(ﻫ) غير مسموح بعلامات الاقتباس .
(و) صحيحة

٢ – ٤٦ (أ) عددى (صحيح) .
(ب) عددى (صحيح) .
(ﺟ) سلسلة حرفية (صحيحة) .
(د) لايمكن أن يتبع علامة الدولار رقم صحيح .
(ﻫ) يجب أن يكون الحرف الأول حرفاً من الحروف الهجائية .
(و) لاتسمح بعض نسخ البيسك أن يتضمن المتغير الحرفى رقماً صحيحاً .
(ز) حروف كثيرة جداً .
(ح) أرقام صحيحة كثيرة جداً .
(ط) حروف هجائية كثيرة جداً .
(ى) سلسلة حرفية (صحيحة) .
(ك) يجب أن يكون الحرف الأول حرفاً من الحروف الهجائية .
(ل) يجب أن يكون ثانى حرف ، إن وجد ، رقماً صحيحاً أو علامة الدولار .
(م) عددى (صحيح) .
(ن) عددى (صحيح) .

٢ – ٤٧ (أ) T↑(N+1)

(ب) (X+3)↑(1/K)

(حـ) 2*(A/B)↑.33333333 or 2*(A/B)↑(1/3)

(د) 1.87*(U+V)−5.088*(X/Y+2*Z↑2)

(هـ) 1−X+X↑2/2−X↑3/6+X↑4/24−X↑5/120

(و) (2*(P/Q)↑(K−1))/((R−3*T)↑(1/M))

(ز) (I+J−1)↑2/5 or 0.2*(I+J−1)↑2

(ح) (((X1+X2)↑M*(Y1+Y2)↑N)/((X1/Y1)↑(M+N)*(X2/Y2)↑(M−N)))↑(1/(M*N))

٢ – ٤٨ يجب أن يتم مسبقاً تحديد قيمة المتغير الذى يظهر على الجانب الأيمن من علامة التساوى فى جملة LET بقيمة عددية أو حرفية مناسبة .

٢ – ٤٩ (أ) 10 LET P=758.33

(ب) 20 LET B=A

(حـ) 30 LET F$="PITTSBURGH, PA."

(د) 40 LET N$=M$

(هـ) 50 LET Y3=X/(A+B−C)

(و) 60 LET K=K−2

(ز) 70 LET C5=2*C5

(ح) 80 LET B=C=(A↑2+B↑2)↑.5

٢ – ٥٠ (أ) 10 LET W=((A+3)*B↑N)/(2.7*(C−D/B)+1)

(ب) 20 LET F=(((A/B)↑N/(C−D)↑M)/(D/(B−A)↑(N+M)))↑(1/(N+M))

(حـ) 30 LET Y=(A1−A2*X+A3*X↑2−A4*X↑3+A5*X↑4)/(C1−C2*X+C3*X↑2−C4*X↑3)

(د) 40 LET P=R*A*(1+R)↑N/((1+R)↑N−1)

٢ – ٥١ (أ)

```
10 LET W1=(A+3)*B↑N
15 LET W2=2.7*(C−D/B)+1
20 LET W=W1/W2
```

(ب)

```
50 LET F1=(A/B)↑N/(C−D)↑M
55 LET F2=D/(B−A)↑(N+M)
60 LET F=(F1/F2)↑(1/(N+M))
```

(حـ)

```
100 LET Y1=A1−A2*X+A3*X↑2−A4*X↑3+A5*X↑4
105 LET Y2=C1−C2*X+C3*X↑2−C4*X↑3
110 LET Y=Y1/Y2
```

(د)

```
200 LET Q=(1+R)↑N
205 LET P=R*A*Q/(Q−1)
```

٢ – ٥٢ (أ) $f = a + 2b/\sqrt{c}$

(ب) $f = a + \sqrt{2b/c}$

(حـ) $f = (a+2)\sqrt{b/c}$

(د) $f = \sqrt{(a+2)b/c}$

(هـ) $g = (pq/r)(s/t)$

٢ – ٥٣ إذا كانت (Y-Z) تمثل كمية سالبة ، إذن فسوف نواجه صعوبة حيث لايمكن رفع الكمية السالبة إلى قوة كسرية فى البسلاء

٢ – ٥٤ P=−(2↑4)=−16

٢ – ٥٥ P=(−2)↑4=16

٢ – ٥٦ (أ)
```
10 INPUT A,B,C,M$,N$
```
(ب)
```
10 INPUT A,N$,B
15 INPUT M$,C
```
(ﺟ)
```
10 INPUT A
12 INPUT B
14 INPUT C
16 INPUT M$
18 INPUT N$
```
(د)
```
10 PRINT "ENTER VALUES FOR A,B,C,M$ AND N$";
20 INPUT A,B,C,M$,N$
```
(ﻫ)
```
10 PRINT "ENTER VALUES FOR A,B,C,M$, AND N$"
20 INPUT A,B,C,M$,N$
```
(و)
```
100 PRINT A,B,C,M$,N$
```
(ز)
```
100 PRINT A;B;C;M$;N$
```
(ح)
```
120 PRINT A;B;C;
```
(ط)
```
200 PRINT A;B;C;(A+B+C)/3;(A*B*C)↑(1/3);(A↑2+B↑2+C↑2)↑.5
210 PRINT
220 PRINT M$,,,,N$
```
(ى)
```
300 PRINT "A=";A,"B=";B,"C=";C
```
أو
```
300 PRINT "A=";A;"B=";B;"C=";C
```
(ك)
```
500 PRINT ,,"NAME: ";M$
510 PRINT
520 PRINT ,,"SOCIAL SECURITY NUMBER: ";N$
```

٢ – ٥٧ (أ)
```
?6.2E−6,27.5E−12,−1000
?SHARON,GAIL
```
يمكن طباعة أول سطر من البيانات أيضاً بالصورة التالية :
```
?.0000062,.275E−10,−1000
```
(ب)
```
?−743.08,.00987,SUSAN
```
أو
```
?−.74308E+3,.987E−2,SUSAN
```
(ﺟ)
```
?"NEW YORK","CHICAGO","SAN FRANCISCO"
```
(علامات الاقتباس مطلوبة حول NEW YORK و SAN FRANCISCO لأن كليهما يتضمن مسافة خالية . أما فى حالة CHICAGO ، فإن علامات الاقتباس تكون اختيارية) .

(د)
```
?2770543,"DECEMBER 29, 1963",48.8E+9,"ELEVEN O' CLOCK"
```
(يجب أن تحصر السلاسل الحرفية بين علامات الاقتباس وذلك لوجود المسافات الخالية والفصلات (,)) .

٢ – ٥٨ (أ)
```
6.20000E-6  2.75000E-11 -1000  2770543 -743.08  9.87000E-3
4.88000E+10
```
(ب)
```
6.20000E-6     2.75000E-11   -1000          2770543       -743.08
9.87000E-3     4.88000E+10
```
(ﺟ)
```
6.17250E-6     5.67734E-5    -75286.7
```
(د)
```
SHARON        DECEMBER 29, 1963          ELEVEN O' CLOCK
```

٢ – ٥٩ (أ)
```
110 REM AVERAGING OF AIR POLLUTION DATA
```
(ب)
```
250 REM BEGIN LOOP TO CALCULATE CUMULATIVE SUM
```
(ﺟ)
```
80 LET A=S/N 'CALCULATE AVERAGE VALUE
```
(د)
```
20 INPUT X,T 'READ A DATA POINT
```

٢ – ٦٠ (أ) صحيحة

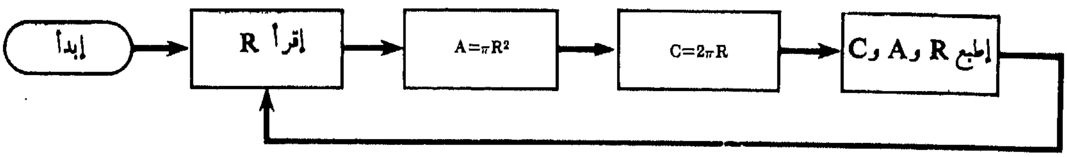
(ب) يجب أن يكون رقم الجملة التى يتحول إليها التحكم رقماً صحيحاً موجباً وليس صيغة .

(ﺟ) لايمكن لجملة GO TO أن تحول التحكم لنفسها .

(د) صحيحة

(ﻫ) لايمكن لعلامات الاقتباس أن تظهر .

٢ – ٦١ خريطة سير العمليات مبينة فى شكل المسألة ٢ – ٦١



شكل المسألة ٢ – ٦١

٢ – ٦٢

(أ)
```
10 REM HELLO!
20 PRINT , , "HELLO!"
30 END
```

(ب)
```
10 REM WELCOME!
20 PRINT "HI, WHAT'S YOUR NAME";
30 INPUT N$
40 PRINT
50 PRINT
60 PRINT "WELCOME "; N$; "!"
70 PRINT "LET'S BE FRIENDS!"
80 END
```

٢ – ٦٣ (أ)
```
10 REM TEMPERATURE CONVERSION PROBLEM
20 PRINT "TEMPERATURE IN DEGREES FAHRENHEIT=";
30 INPUT F
40 LET C=5*(F-32)/9
50 PRINT "DEGREES F=";F,"DEGREES C=";C
60 END
```

(ب)

```
10 REM PIGGY-BANK PROBLEM
20 PRINT "NUMBER OF HALF-DOLLARS=";
30 INPUT N1
40 PRINT "NUMBER OF QUARTERS=";
50 INPUT N2
60 PRINT "NUMBER OF DIMES=";
70 INPUT N3
80 PRINT "NUMBER OF NICKELS=";
90 INPUT N4
100 PRINT "NUMBER OF PENNIES=";
110 INPUT N5
120 LET S=.5*N1+.25*N2+.1*N3+.05*N4+.01*N5
130 PRINT "TOTAL AMOUNT OF MONEY=";S;" DOLLARS"
140 END
```

لاحظ أن جملة END يمكن أن تستبدل بجملة GO TO ملائمة فى كل من المسائل الموضحة عاليه . وسوف يسمح ذلك لكل برنامج بمعالجة مجموعات متعددة من البيانات بالتعاقب .

٢ - ٦٤ فيما يلى برنامج بيسك كامل لكل مسألة ومع ذلك ، يجب أن يكون مفهوماً أن البرمجة الحقيقية لايجب أن تبدأ حتى يتم تجهيز مخطط تمهيدى أو خريطة سير عمليات . كل برنامج مكتوب بطريقة تسمح بمعالجة مجموعات عديدة من البيانات فى صورة متعاقبة .

(أ)

```
10 REM VOLUME AND AREA OF A SPHERE
20 LET P=3.1415927
30 PRINT "RADIUS=";
40 INPUT R
50 LET V=4*P*R↑3/3
60 LET A=4*P*R↑2
70 PRINT "R=";R,"V=";V,"A=";A
80 PRINT
90 GO TO 30
100 END
```

(ب)

```
10 REM COMPUTATION OF MASS OF AIR IN A TIRE
20 PRINT "P=";
30 INPUT P
40 PRINT "V=";
50 INPUT V
60 PRINT "T=";
70 INPUT T
80 LET M=P*V/(.37*(T+460))
90 PRINT "M=";M
100 PRINT
110 GO TO 20
120 END
```

(جـ)

```
10 REM GEOMETRIC PROPERTIES OF A TRIANGLE
20 PRINT "A=";
30 INPUT A
40 PRINT "B=";
50 INPUT B
60 PRINT "C=";
70 INPUT C
80 LET S=(A+B+C)/2
90 LET A0=(S*(S—A)*(S—B)*(S—C))↑.5
100 LET R1=A0/S
```

```
110 LET A1=3.14159*R1↑2
120 LET R2=A*B*C/(4*A0)
130 LET A2=3.14159*R2↑2
140 PRINT "AREA OF TRIANGLE=";A0
150 PRINT "AREA OF LARGEST INSCRIBED CIRCLE=";A1
160 PRINT "AREA OF SMALLEST CIRCUMSCRIBED CIRCLE=";A2
170 PRINT
180 GO TO 20
190 END
```

(د)

```
10 REM COMPOUND INTEREST PROBLEM
20 PRINT "P=";
30 INPUT P
40 PRINT "I=";
50 INPUT I
60 PRINT "N=";
70 INPUT N
80 LET F=P*(1+I)↑N
90 PRINT "F="; F
100 PRINT
110 GO TO 20
120 END
```

إذا تضاعف الربح كل ربع سنة وليس سنوياً فيجب تغير الجملة رقم 80 لتقرأ كما يلي :

```
80 LET A=P*(1+I/4)↑(4*N)
```

(هـ)

```
10 REM GROWTH OF A BACTERIA POPULATION
20 PRINT "T=";
30 INPUT T
40 LET C=.0289*T
50 LET F=1+C+C↑2/2+C↑3/6+C↑4/24+C↑5/120+C↑6/720+C↑7/5040+C↑8/40320
60 LET F=F+C↑9/362880
70 PRINT "P/P0=";F
80 PRINT
90 GO TO 20
100 END
```

لاحظ أن حساب حاصل الضرب المعامل (F) يتطلب جملتين حيث أن جملة واحدة يمكن أن تتعدى طول السطر (يتوقف ذلك على طولها) .

٤ - ٣٢ (أ) صحيحة

(ب) لايمكن لهذا الشرط أن يستوفى أبداً .

(حـ) صحيحة .

(د) صحيحة .

(هـ) صحيحة .

(و) لايمكن مقارنة متغير رقمي بآخر حرفي .

(ز) صحيحة .

٤ – ٣٣ يجب أن تأتى السلسلة الحرفية الممثلة بواسطة P$ قبل السلسلة الحرفية الممثلة بواسطة Q$ وذلك كترتيب الهجائى .

٤ – ٣٤ (أ) صحيحة

(ب) غير صحيحة لغوياً وهيكلياً (يمكن أن يتبع THEN رقم جملة فقط) .

(ﺣ) لاتسمح كل نسخ البيسك باستخدام GO TO بدلا من THEN .

(د) صحيحة .

(ﻫ) يجب أن يكون رقم الجملة التى يتحول إليها التحكم رقماً صحيحاً موجباً وليس متغيراً .

(و) صحيحة .

(ز) لايمكن أن يستوفى الشرط على الإطلاق .

٤ – ٣٥ (أ)

```
40 IF K<15 THEN 50
```

(ب)

```
100 IF N$="OPTION A" THEN 70
110 GO TO 150
```

(ﺣ)

```
 60 ...
 ...
150 IF X>=100 THEN 200
160 LET J=J+1
170 INPUT X
180 GO TO 60
 ...
200 ...
```

(د)

```
 20 ...
 ...
 80 IF J=0 THEN 150
 90 LET S=S+J
100 GO TO 20
 ...
150 ...
```

٤ – ٣٦ (أ) صحيحة

(ب) سوف تستمر الحلقة التكرارية إلى مالا نهاية لأن قيمة J سوف تكون دائماً I . (إذا تغيرت الجملة 80 إلى

```
80 GO TO 40
```

فسوف تنفذ الحلقة التكرارية بصورة صحيحة) .

(ﺣ) سوف يتحول التحكم دائماً إلى الجملة رقم 200 .

(د) سوف تستمر الحلقة التكرارية إلى مالا نهاية ، حيث لن تتعدى قيمة X الرقم 100 على الإطلاق .

(ﻫ) صحيحة

٤ – ٣٧ (أ) لايمكن أن يظهر متغير حرفى فى جملة ON-GO TO .

(ب) صحيحة

(ﺣ) لايمكن لجملة ON–GO TO أن تحول التحكم لنفسها .

(د) يجب أن يكون رقم الجملة التى يتحول إليها التحكم رقماً صحيحاً موجباً . وليس متغيراً .

(ﻫ) صحيحة

(و) لاتسمح كل نسخ البيسك باستخدام THEN بدلا من GO TO .

(ز) غير صحيحة فى الهيكل اللغوى (يجب أن تسبق ON الجملة GO TO)

٤ – ٣٨ (أ) سوف تنتج رسالة خطأ (لا يمكن أن نواصل التنفيذ لأن 1 – = J–K) .
(ب) سوف يتحول التحكم إلى الجملة رقم 20 .
(حـ) سوف يتحول التحكم إلى الجملة رقم 20.
(د) سوف تنتج رسالة خطأ (لايمكن أن نواصل التنفيذ لأن 5 = J–K) .
(هـ) سوف يتحول التحكم إلى الجملة رقم 100 .

٤ – ٣٩ (أ) صحيحة
(ب) لايمكن للصيغة الرياضية أن تظهر مكان المتغير الجارى .
(حـ) لايمكن للمتغيرات الحرفية أن تظهر فى جملة FOR–TO .
(د) صحيحة (لاحظ أن V (1) وV (2) و V (3) كلها متغيرات ذات أدلة ، حيث تمت مناقشتها فى الفصل 5) .
(هـ) صحيحة
(و) جملة غير منطقية (مطلوب أن نقلل من قيمة المتغير الجارى حتى يتناقص ولكننا لانستطيع عمل ذلك لأن حجم الخطوة STEP قيمة موجبة) .

٤ – ٤٠ (أ)

```
10 FOR I=1 TO 200
...
70 NEXT I
```

توضع أرقام الجمل اختيارياً

(ب)

```
10 FOR I=1 TO 200
     ...
50      IF X<.001 THEN 175
     ...
70 NEXT I
...
175 ...
```

(حـ)

```
10 FOR I=1 TO 73 STEP 3
...
70 NEXT I
```

(د)

```
10 FOR C=.5 TO A↑3–10 STEP A+B
...
70 NEXT C
```

٤ – ٤١ (أ) تتغير قيمة المتغير الجارى بداخل الحلقة التكرارية .
(ب) صحيحة
(حـ) تتداخل الحلقات التكرارية بصورة متشابكة .
(د) تستخدم الحلقة التكرارية الداخلية والخارجية نفس المتغير الجارى (X) . وكذلك فإن كلتا الحلقتين أيضاً تنتهى عند جملة NEXT واحدة .
(هـ) صحيحة طالما تم تحديد قيم عددية للمتغيرين T و T1 .
(و) أن المتغير الجارى (C) فى الجملة NEXT ليس هو نفس المتغير (X) فى جملة FOR–TO .

٥ – ٤٠

(أ) `10 LET Y=SQR(SIN(X)–COS(X))`

(ب) `10 LET P=Q*EXP(–Q*T)`

(حـ) `10 LET C=LOG(SQR(ABS(A+B)))+LOG(SQR(ABS(A–B)))`

(د) `10 LET W=ABS(ABS(U–V)–ABS(U+V))`

(هـ) `10 LET Z=COS(X+ATN(Y))`

٥ – ٤١ (أ)

```
10 ON SGN((A*B-C*D)/(F+G))+2 GO TO 135,260,75
```

(ب)

```
100 PRINT TAB(4);"X=";X;TAB(28);"Y=";Y;TAB(52);"Z=";Z
```

(ﺣ)

```
100 IF (N/2)=INT(N/2) THEN 200
```

سوف يتحول التحكم إلى الجملة رقم 200 إذا كانت قيم N زوجية .

(د) سوف يستمر تحويل التحكم إلى الجملة رقم 200 إذا كانت قيم N زوجية .

٥ – ٤٢ (أ) N$ هى قائمة حرفية .

(ب) A عبارة عن جدول رقمى و A$ عبارة عن جدول حرفى و C قائمة رقمية و C$ قائمة حرفية .

(ﺣ) P$ قائمة حرفية و P جدول رقمى .

(د) Z جدول رقمى .

٥ – ٤٣ (أ) صحيحة

(ب) صحيحة

(ﺣ) يجب أن تكون الكية المحصورة بين القوسين فى جملة DIM كية صحيحة موجبة ، وغير مسموح باستعمال المتغيرات .

(د) لايمكن أن تحتوى جملة DIM متغيرات بدون أدلة (فى هذه الحالة المتغيرات C1 و C) .

(ﻫ) لايمكن أن يكون الدليل قيمة سالبة .

(و) صحيحة

٥ – ٤٤ (أ)

```
10 LET S=0
20 FOR I=1 TO 199 STEP 2
30     LET S=S+X(I)↑2
40 NEXT I
50 LET S1=SQR(S)
```

(ب)

```
10 FOR I=1 TO 8
20     FOR J=1 TO 12
30         LET H(I,J)=1/(I+J-1)
40     NEXT J
50 NEXT I
```

(ﺣ)

```
100 FOR I=1 TO N
110     IF K(I)>15 THEN 130
120     PRINT TAB(8);"I=";I;TAB(44);"K=";K(I)
130 NEXT I
```

(د)

```
100 LET P=1
110 FOR I=1 TO K
120     LET P=P*W(I,I)
130 NEXT I
```

(ﻫ)

```
100 FOR I=1 TO M
110     PRINT TAB(10);M$(I,4)
120 NEXT I
```

(و)

```
100 FOR I=1 TO M
110     PRINT M$(I,4);" ";
120 NEXT I
```

(ز)

```
100 FOR J=1 TO N
110     PRINT M$(5,J);" ";
120 NEXT J
```

٥ - ٤٥ (أ)

```
10 READ L$(1),L$(2),L$(3),L$(4),P,Q,R,H$
20 READ T(1,1),T(1,2),T(1,3),T(1,4),T(2,1),T(2,2),T(2,3),T(2,4)
...
200 DATA WHITE, YELLOW, ORANGE, RED
210 DATA 2.25E+5,6.08E−9,−1.29E+12,RESTART
220 DATA 1,−3,5,−7,−2,4,−6,8
```

يمكن لجمل READ و DATA أن تضم أو يتم توسيعها وذلك حسب الرغبة .

(ب)

```
10 FOR I=1 TO 4
20     READ L$(I)
30 NEXT I
40 READ P,Q,R,H$
50 FOR I=1 TO 2
60     FOR J=1 TO 4
70         READ T(I,J)
80     NEXT J
90 NEXT I
```

جمل DATA سوف تكون هى نفسها الموجودة فى الجزء (أ) .

(ﺣ)

```
10 FOR I=1 TO 4
20     READ L$(I)
30 NEXT I
40 READ P,Q,R,H$
50 FOR I=1 TO 2
60     FOR J=1 TO 4
70         READ T(I,J)
80     NEXT J
90 NEXT I
...
150 RESTORE*
160 READ P1,Q1,R1
```

جمل DATA سوف تكون هى نفسها كما فى الجزء (أ .

(د) الحل هو نفسه الحل فى الجزء (ﺣ) ماعدا أن الجملة رقم 150 يجب أن تستبدل إما بالجملة ،

```
150 RESTORE
```

أو بالجملة

```
150 RESTORE$
```

يجب أن تستبدل الجملة رقم 160 بالجملة :

```
160 READ A1$,A2$,A3$,A4$
```

٦ - ٤١ (أ)

```
10 DEF FNY(A,B,X)=A*X↑B
```

(ب)

```
10 DEF FNQ(R)=C0+C1*R+C2*R↑2+C3*R↑3+C4*R↑4
```

يجب أن تعرف قيم المتغيرات C0 إلى C4 فى مكان آخر من البرنامج ، قبل الرجوع والإشارة إلى الدالة .

(ﺣ)

```
10 DEF FNI(J,K)=(J+K)↑(J+K)
```

(د)

```
10 DEF FNR(A,B,C)
20 IF B↑2<4*A*C THEN 50
30 LET FNR=SQR(B↑2−4*A*C)
40 GO TO 60
50 LET FNR=SQR(4*A*C−B↑2)
60 FNEND
```

٦ – ٤٢ (أ)

```
10 DEF FNZ(Z)=INT(Z+.5)
```

(ب)

```
10 DEF FNP(N)
20 LET P=1
30 FOR I=1 TO N
40     LET P=P*T(I)
50 NEXT I
60 LET FNP=P
70 FNEND
```

(جـ)

```
10 DEF FNR(A,B)
20 LET FNR=0
30 FOR I=1 TO 5
40     LET X=A+(B-A)*RND
50     IF X<=R THEN 70
60     LET FNR=X
70 NEXT I
80 FNEND
```

(د)

```
10 DEF FNW$(X)
20 ON SGN(X)+2 GO TO 30,50,70
30 LET FNW$="NEGATIVE"
40 GO TO 80
50 LET FNW$="ZERO"
60 GO TO 80
70 LET FNW$="POSITIVE"
80 FNEND
```

(هـ)

```
10 DEF FNL$(N$)
20 CHANGE N$ TO L
30 LET N=90
40 FOR I=1 TO L(0)
50     IF N<=L(I) THEN 70
60     LET N=L(I)
70 NEXT I
80 LET FNL$=CHR$(N)
90 FNEND
```

لاحظ أن المقدار الثابت 90 يساوى الحرف Z فى كود ASCII (أنظر الجدول ٦ – ١)

٦ – ٤٣ (أ)

```
100 LET T=FNY(C1+C2,3,X+Y)
```

(أنظر الحل المعطى للمسألة ٦ – ٤١ (أ)) .

(ب)

```
100 LET Q=FNQ(LOG(X))
```

(أنظر الحل المعطى للمسألة ٦ – ٤١ (ب)) .

(جـ)

```
100 PRINT FNI(A-B,C)
```

(أنظر الحل المعطى للمسألة ٦ – ٤١ (جـ)) .

(د)

```
100 LET D=ABS(X-FNZ(X))
```

(أنظر الحل المعطى للمسألة ٦ – ٤٢ (أ)) .

(هـ)

```
10 FOR I=1 TO 60
20     IF FNP(I)>1000 THEN 40
30 NEXT I
40 PRINT "N"; I
```

(أنظر الحل المعطى للمسألة ٦ – ٤٢ (ب)) .

(و)

```
10 FOR I=1 TO 20
20     PRINT "I=";I,"LARGEST R=";FNR(2,5)
30 NEXT I
```

(أنظر الحل المعطى للمسألة ٦ – ٤٢ (جـ)) .

٦ – ٤٤ (أ) عدد الخلاصات الموجودة فى الإشارة للدالة لايتوافق مع عدد الخلاصات الموجودة فى تعريف الدالة .

(ب) يحتوى هذا المثال على خطأين :

(i) لم يعط إسم الدالة (FNC) قيمة من خلال الدالة .

(ii) لايمكن لجملة RETURN أن تظهر فى تعريف الدالة .

(ـه) يحتوى هذا المثال على خطأين :

(i) لايمكن أن يتحول التحكم إلى البرنامج الصغير الفرعى بواسطة جملة GO TO .

(ii) لايمكن أن تظهر جملة FNEND فى برنامج صغير فرعى .

(د) يحتوى هذا المثال على خطأين :

(i) أعط إسم غير صحيح للدالة

(ii) لايمكن أن تظهر الصيغ الرياضية كخلاصات زائفة .

(ﻫ) الخلاصات غير موجودة فى المرجع إلى الدالة .

(و) لايمكن تحويل التحكم إلى خارج برنامج صغير فرعى بواسطة جملة IF-THEN .

(ز) منطق البرنامج غير صحيح . (تشير الدالة إلى البرنامج الصغير الفرعى ، وهو بدوره يشير إلى الدالة) .

٧ – ٣٩ (أ) صحيحة

(ب) لايمكن أن تظهر الصيغ الرياضية الخاصة بالمصفوفات فى جمل المصفوفات .

(ـه) صحيحة .

(د) صحيحة .

(ﻫ) صحيحة .

(و) لايمكن تعديل المصفوفة بواسطة ضرب المصفوفة .

(ز) لايمكن القيام بعملية ضرب المصفوفة إلا إذا كان عدد أعمدة المصفوفة الأولى (X) هو نفسه عدد صفوف المصفوفة الثانية (Y) . وفى هذا المثال فإن المصفوفة (X) لها 20 عموداً ولكن المصفوفة (Y) لها عشرة صفوف فقط .

(ح) يجب أن تكتب MAT INPUT بالصورة :

```
50 MAT INPUT V
```

(ط) صحيحة ، طالما يتم تحديدM و N بقيم صحيحة موجبة لاتتعدى 100 و50 بالترتيب .

(ى) لايمكن استخدام خاصية البعد المتغير مع جملة MAT PRINT .

(ك) صحيحة .

(ل) يتعدى أحد الأبعاد الموصفة فى جملة MAT ZER البعد المناظر له فى جملة DIM .

(م) لايمكن الحصول على مقلوب مصفوفة غير مربعة .

(ن) صحيحة .

٧ – ٤٠ (أ)

```
100 MAT I=IDN
110 MAT B=TRN(A)
120 MAT C=B*A
130 MAT D=C-I
140 MAT F=(2*N+1)*D
```

سوف تكون F مصفوفة لها الأبعاد 10×10

(ب)

```
100 MAT E=A*C
110 MAT F=B*D
120 MAT G=E-F
130 MAT H=INV(G)
140 LET D1=DET
150 PRINT "DETERMINANT OF G=";D.
```

(ﺟ) اجمع الجملة :

```
145 MAT PRINT G;H;
```

إلى الجزء (ب) السابق .

(د)

```
100 MAT A=ZER(12,8)
```

(ﻫ)

```
100 MAT READ A(8,12),B(6,15)
```

يجب أن تحتوى كتلة البيانات على عناصر المصفوفة ، بترتيب الصفوف A (أى الصف الأول ثم الثانى ... وهكذا) وتتبع هذه القيم قيم المصفوفة B بترتيب الصفوف أيضاً .

(و)

```
100 FOR I=1 TO 8
110     FOR J=1 TO 12
120         PRINT A(I,J);
130     NEXT J
140     PRINT
150 NEXT I
160 PRINT
170 FOR I=1 TO 6
180     FOR J=1 TO 15
190         PRINT B(I,J);
200     NEXT J
210     PRINT
220 NEXT I
```

الحلقات التكرارية FOR-TO مطلوبة لأن خاصية البعد المتغير غير متاحة مع جملة MAT PRINT .

٨ - ٣٤ (أ) تستخدم معظم نسخ البيسك جملة INPUT بدلا من جملة READ عند قراءة ملف بيانات متتال .

(ب) لايمكن أن يحتوى ملف بيانات عشوائى على ثوابت حرفية ورقية معاً .

(ﺟ) صحيحة طالما تم تحديد ملفات البيانات العشوائية لقنوات البيانات 1 و 2 .

(د) صحيحة .

(ﻫ) يجب أن تتضمن جمل SCRATCH و QUOTE أرقام قناة بيانات وليس أسماء ملفات .

(و) يجب أن تتضمن جملة FILES إسم ملف وليس متغيراً حرفياً .

(ز) لايمكن وضع المؤشر وراء نطاق نهاية الملف .

(ح) لايمكن كتابة سلسلة حرفية على ملف بيانات عشوائى.

٨ - ٣٥ (أ)

```
10 FILES LIST1,,LIST2
```

(ب)

```
10 FILES NAMES$25,ACCTS%
```

(ﺟ)

```
10 FILE :2,F$
20 FILE :5,G$
```

(د)

```
NEW OR OLD-->NEW
NEW FILE NAME-->TAPE1
10 ...
20 ...
...            TAPE 1 ملف البيانات
300 ...
SAVE
LIST
```

(هـ)

```
NEW OR OLD-->NEW
NEW FILE NAME-->ITEMS
SAVE
```

(و)

```
OLD
OLD FILE NAME-->OLD1
SCRATCH
OLD
OLD FILE NAME-->OLD2
SCRATCH
OLD
OLD FILE NAME-->NEW1
RENAME OLD1
OLD
OLD FILE NAME-->NEW2
RENAME OLD2
```

(ز) محتمل أن البرنامج الموصوف فى المسألة ٨ - ٣٥ (و) سوف يحتوى على الجملة :

```
10 FILES OLD1,OLD2,NEW1,NEW2
```

يجب أن تستبدل الجملة بسلسلة من الجمل التالية :

```
10 INPUT A$,B$,C$,D$
20 FILE #1,A$
30 FILE #2,B$
40 FILE #3,C$
50 FILE #4,D$
```

عند تنفيذ البرنامج فإن الأسماء الموجودة فى ملفات الإدخال سوف تعطى للمتغيرات A$ و B$ ، أما الأسماء الموجودة فى ملفات الإخراج فسوف تعطى للمتغيرات C$ و D$.

(ح)

```
10 INPUT A$,B$
20 FILE #5,A$
30 FILE #3,B$
40 QUOTE #3
50 SCRATCH #3
60 INPUT #5,N,F$,X,Y,Z,G$
70 PRINT #3,N,Z,F$,G$
80 IF END #5, THEN 100
90 GO TO 60
100 END
```

(ط)

```
10 INPUT A$,B$
20 FILE :5,A$
30 FILE :3,B$
40 READ :5,X
50 WRITE :3,X
60 IF LOC(5)=LOF(5) THEN 80
70 GO TO 40
80 END
```

(ى)

```
50 INPUT P
60 SET :6,P
70 READ :6,X
80 SET :2,P
90 WRITE :2,X
```

(ك)

```
100 LET P1=LOC(1)
110 LET P4=LOC(4)
120 IF P1=P4 THEN 200
130 LET P2=P1
140 IF P4<=P1 THEN 160
150 LET P2=P4
160 SET :2,P2
```

(ل)

```
100 IF LOF(3)<>LOF(5) THEN 25
```

المصطلحات العلمية (عربى — انجليزى)

(أ)

اجراء	Procedure
اجراء التسجيل للخروج من النظام	Logout procedure
اجراء التسجيل لدخول النظام	Login procedure
اجراء التكرار	Iterative procedure
الاجراء الحسابى	Computational procedure
الاخطاء اللغوية	Grammatical errors
ادماج	Merge
ارتداد كرة	Bouncing ball
الأرقام الأولية	Primary numbers
ارقام عشوائية كاذبة	Pseudo-random numbers
ارقام فيبوناسى	Fibonacci numbers
ارقام معنوية	Significant figures
ارتداد كرة	Bouncing ball
ارقام صحيحة	Integers
ارقام حقيقية	Reals
أسلوب	Mode
اساليب التشغيل	Modes of operation
الاستكمال	Interpolation
استهل	Initialize
استهلاك	Depreciation
الاسواق المالية	Stock market
الاشارة او الرجوع	Reference
اعادة ترتيب قائمة	Reordering a list
افتراضيا	Virtually
اكتشاف الخطأ وتصحيحه	Debugging
أماكن خالية	Blank spaces
انحدار خطى	Linear regression
انحراف	Deviation
الانحراف المعيارى	Standard deviation
أولى	Elementary

(ب)

بحث ثنائى	Binary search

In terms of	بدلالـــة
Source program	برنامج المنبع
Object program	برنامج الهدف
Structured program	برنامج هيكلى
Skeletal structure	بنـــــاء هيكلى
Items	بنـــود
High resolution graphics	بيانيات عالية الوضـوح
Enhanced Basic	البيسك المحسن

(ت)

Spacing data items	تباعد بنود البيانات
Variance	التباين
Tracing	التتبـــع
Debugging	تحديد الأخطاء
Transform	تحويـــل
User-ID	تحقيق شخصية المستفيد
Transferring control	تحويل التحكم
Handling files	تداول الملفات
Hierarchy of operations	التدرج الهرمى للعمليات
Transpose	تدوير
Transpose of a matrix	تدوير مصفوفة
Compilation	ترجمة البرامج
Sequence	تسلسل
Error diagnostics	تشخيص الأخطاء
Processing a program	تشغيل برنامج
Manipulating files	التعامل مع الملفات
Instructions	تعليمات
Relational expression	تعبير مترابط
Successive substitution	التعويض المتتالى
Interpretation	تفسير البرامج
Encoding	تكويد شفرة
Prompting	تـــلقين
Air pollution	تلوث الهواء
Editing	تنقيـــح
Access	توصـــل
Direct access	التوصل المباشر
Curve fitting	توفيق منحنى

Generating random numbers	توليد ارقام عشوائية

(ج)

Table of variables	جدول من المتغيرات
Time sharing session	جلسة المشاركة الزمنية

(ح)

Microcomputers	الحاسبات الدقيقة
Commulative product	حاصل الضرب التراكمى
Data field	حقل بيانات
Nested loops	حلقات تكرارية متداخلة
Loop	حلقة تكرارية

(خ)

Actual arguments	خلاصات حقيقية
Dummy arguments	خلاصات زائفة
Argument	خلاصة

(د)

String function	دالة حروف
Statement function	دالة لها جملة واحدة
Multiline function	دالة متعددة الأسطر
Loop index	دليل الحلقة التكرارية
Built-in functions	دوال مبنية داخليا
Library functions	دوال مكتبية
Cycle	دورة

(ر)

Carriage return	رجوع العربة
Error message	رسالة خطأ

(س)

Data record	سجل بيانات
Horizontal velocity	السرعة الأفقية
String	سلسلة الحروف

(ش)

Monitor	شاشة عرض

(ص)

Hard-copy graph	صورة بيانية على ورق طباعة
Formula	صيغة رياضية

(ط)

Line printer	طابع السطور
Computational procedure	طريقة اجراء الحسابات
Overflow	طفح
Least squares method	طريقة اقل المربعات
Conversational manner	طريقة تخاطبية

(ع)

Quotation Marks	علامات الاقتباس
Colon (:)	علامة الوقف الاستدراكى
Customers	عملاء

(غ)

Unformatted	غير مصاغة
Unscrambling	غير مرئية

(ف)

Unique	فريد
Decoding	فك شفرة
Fibonacci	فيبوناسى

(ق)

List of variables	قائمة من المتغيرات
Data Channel	قناة البيانات
Rules of the game	قواعد اللعبة
Initial value	القيمة الابتدائية
Final value	القيمة النهائية
Median	القيمة الوسيطة

(ك)

Data block	كتلة البيانات
Fraction	كسر
Key words	الكلمات الدالة
Console	كونسول
Entity	كيان

(ل)

لذلك	Hence
لغة الآلة	Machine language
لغة رفيعة المستوى	High level language
اللفة الراسية	Vertical scrolling

(م)

مبرمج	Programmer
متتابعة	Sequential
متسلسلة	Series
متضاعفة الدقة	Double precision
متعددة	Versatility
المتغير الجارى	Running variable
متغيرات ذات ادلة	Subscripted variables
المتوسط المحسوب	Calcualted average
المتوسط المرجح	Weighted average
المتوسط الهندسى	Geometric average
مجموعة متراصة	Array
محاكاة	Simulation
المحاولة والخطأ	Trial-and-error
المخطط التمهيدى للبرنامج	Program outline
مخطط الحذف	Elimination scheme
المخرجات البيانية	Graphical output
مراقبة المخزون	Inventory control
المرتبات الأسبوعية	Weekly payroll
مركز الشاشة	Center of the screen
مزدوجة الدقة	Double Precision
المساحة تحت المنحنى	Area under the curve
مستوى المخزون	Inventory level
مصاغة	formatted
مصفوفة	Matrix
معالجة الكلمات	Word processing
معامل الترابط	Relational operator
معاملات	Parameters
المعاملات المنطقية	Logical operators
معاملة	Transaction
مقدار الخطوة	Step value

مقدار الزيادة	Increment
الملحوظـة	Remark
ملازم — مقترن	Associated
ملف البيانات العشوائى	Random data file
ملف متتابع	Sequential file
مـوردون	Suppliers
موزع بانتظام	Uniformally distributed
موزع طبيعيا	Normally distributed
مؤشر	Pointer
مؤشر وامض	Cursor
موضـع	Location
مولد الرقم العشـوائى	Random number generator

(ن)

نـص	Text
نطاق البحث	Search interval
النهاية الطرفية المركزية	Terminal console
نهـاية مـلف	End of file

(هـ)

هيستوجرام	Histogram

(و)

الوصـل	Concatenation

(ى)

يبتـر	Truncate
يتذبذب	Fluctuate
يحـذف	Delete
يحدد قيمة	Assign
يرمز او يكود	Encode
يستدعى	Invoke
يفك الشفرة	Decode

Round-up	يقرب
Cancel	يلغى
Scratch	يمسح
Utilize	ينتفع
Accomplish	ينجز

الفهرس الأبجدى

ملحق (أ)

ملخص لجمل BASIC القياسية

جملة	مثال	مرجع
CHANGE	10 CHANGE N$ TO N	قسم ٦ — ٤
DATA	10 DATA 12,SEVENTEEN,−5	قسم ٥ — ٥
DEF	10 DEF FNR(A,B,C)=SQR(A↑2+B↑2+C↑2)	قسم ٦ — ١
DIM	10 DIM A(10,20),X(20),F$(60)	قسم ٥ — ٤
END	10 END	قسم ٢ — ١١
FNEND	10 FNEND	قسم ٦ — ٣
FOR-TO	10 FOR J=1 TO 99 STEP 2	قسم ٥ — ٤
GO TO	10 GO TO 50	قسم ٢ — ١٤
GOSUB	10 GOSUB 300	قسم ٦ — ٩
IF-THEN	10 IF I>=100 THEN 80	قسم ٤ — ٢
INPUT	10 INPUT A,B,C,M$,N$	قسم ٢ — ٩
LET	10 LET A=3.141593*R↑2	قسم ٢ — ٨
NEXT	10 NEXT I	قسم ٤ — ٦
ON-GO TO	10 ON K GO TO 15,40,25,40,60	قسم ٤ — ٣
PRINT	10 PRINT "X=";X,"Y=";Y	قسم ٢ — ١٠
RANDOMIZE	10 RANDOMIZE	قسم ٦ — ٧
READ	10 READ K,N$,Z(1)	قسم ٥ — ٥
REM	10 REM AREA OF A CIRCLE	قسم ٢ — ١٣
RESTORE	10 RESTORE	قسم ٥ — ٦
RETURN	10 RETURN	قسم ٦ — ٨
STOP	10 STOP	قسم ٤ — ٤
MAT =	10 MAT C=A	قسم ٧ — ١
MAT +	10 MAT C=A+B	قسم ٧ — ١
MAT −	10 MAT C=A−B	قسم ٧ — ١
MAT (K)*	10 MAT C=(10)*A	قسم ٧ — ١
MAT *	10 MAT C=A*B	قسم ٧ — ١
MAT CON	10 MAT B=CON	قسم ٧ — ٣
MAT IDN	10 MAT C=IDN	قسم ٧ — ٣
MAT INPUT	10 MAT INPUT A	قسم ٧ — ٢
MAT INV	10 MAT B=INV(A)	قسم ٧ — ٣
MAT PRINT	10 MAT PRINT A	قسم ٧ — ٢

جملة	مثال	مرجع
MAT READ	10 MAT READ A	قسم ٧ — ٢
MAT TRN	10 MAT B=TRN(A)	قسم ٧ — ٣
MAT ZER	10 MAT A=ZER	قسم ٧ — ٣
FILE	10 FILE :1,F$	قسم ٨ — ٣
FILES	10 FILES SCORES	قسم ٨ — ١
IF END-THEN	10 IF END #1, THEN 130	قسم ٨ — ١
INPUT	10 INPUT #1,N,T$,Y$	قسم ٨ — ١
PRINT	10 PRINT #2,N;N$	قسم ٨ — ١
QUOTE	10 QUOTE #2	قسم ٨ — ١
READ	10 READ :1,L	قسم ٨ — ٢
SCRATCH	10 SCRATCH #2	قسم ٨ — ١
SET (RESET)	10 SET :1,L	قسم ٨ — ٢
WRITE	10 WRITE :1,N	قسم ٨ — ٢

المعاملات الرياضية	↑	/	*	−	+	
المعاملات الرابطة	>	>=	<	<=	<>	=

ملحوظة : ↑ يظهر ك ^ على بعض أو فى بعض الشاشات)

ملحق (ب)

ملخص للدوال المكتبية القياسية فى لغة البيسك

دالـــة	مثـــال	مرجـــع
ABS	10 LET Y=ABS(X)	قسم ٥ — ١
ATN	10 LET Y=ATN(X)	قسم ٥ — ١
ASC	10 LET N=ASC(T)	قسم ٦ — ٥
CHR$	10 LET N$=CHR$(N)	قسم ٦ — ٥
COS	10 LET Y=COS(X)	قسم ٥ — ١
COT	10 LET Y=COT(X)	قسم ٥ — ١
DET	10 LET X=DET	قسم ٧ — ٣
EXP	10 LET Y=EXP(X)	قسم ٥ — ١
INT	10 LET Y=INT(X)	قسم ٥ — ١
LOC	10 LET N=LOC(1)	قسم ٨ — ٢
LOF	10 LET N1=LOF(3)	قسم ٨ — ٢
LOG	10 LET Y=LOG(X)	قسم ٥ — ١
NUM	10 LET N(0)=NUM	قسم ٧ — ٢
RND	10 LET X=RND	قسم ٦ — ٦
SGN	10 LET Y=SGN(X)	قسم ٥ — ١
SIN	10 LET Y=SIN(X)	قسم ٥ — ١
SQR	10 LET Y=SQR(X)	قسم ٥ — ١
TAB	10 PRINT TAB(N);X	قسم ٥ — ١
TAN	10 LET Y=TAN(X)	قسم ٥ — ١

ملحق (ج)

ملخص لأوامر النظام القياسية فى لغة البيسك

الأمر	الغرض
BYE	ينهى جلسة المشاركة الزمنيـــة .
CATALOG	يعطى قائمة باسماء كل الملفات التى تم الاحتفاظ بها .
GOODBYE	تماما كامر BYE
LIST	يعطى قائمة بالملف الحالى .
NEW	يصف ان ملفا جديدا سوف ينشا .
OLD	يتوصل الى ملف سبق الاحتفاظ به
RENAME	يسمح بتغير اسم الملف الحالى .
REPLACE	يتسبب الاحتفاظ بالملف الحالى (تخزينه) فى مكان ملف سبق تخزينه بنفس الاســم . (سوف يلغى الملف القديم) .
RUN	يتسبب فى ترجمة وتنفيذ البرنامج الحالى .
SAVE	يتسبب فى الاحتفاظ بالملف الحالى (تخزينه)
SCRATCH	يزيل الملف الحالى من ذاكرة الحاسب .
SYSTEM	يحول التحكم من البيسك الى مراقب النظام .
UNSAVE	يلغى ملفا تم تخزينه فى المخزن الدائـــم .

ملحق (د)

Summary of Microsoft BASIC

Statements

Statement	*Purpose*	*Example*
BEEP	Beeps the speaker	10 BEEP
BLOAD	Loads a binary memory image	10 BLOAD "SAMPLE"
BSAVE	Saves a binary memory image	10 BSAVE "SAMPLE",0,&H8000
CALL	Calls a machine language subroutine	10 CALL START
CHAIN	Passes control to another program	10 CHAIN "PROGRAM2"
CIRCLE	Generates circles, arcs and ellipses (graphics mode)	10 CIRCLE (160,100),30,2
CLOSE	Closes a file for input/output operations	10 CLOSE #1
CLS	Clears the screen and "homes" the cursor	10 CLS
COLOR	Sets colors or other screen attributes	10 COLOR 7,0,0
COMMON	Defines a common storage area, for passing variables to a chained program	10 COMMON A,B,C,T$
DATA	Provides values for variables listed in READ statement	10 READ A,B,C,T$ 20 DATA 2.3,−0.1,6,RED
DATE$	Sets the date	10 DATE$="12/29/82"
DEF FN...	Defines a function	10 DEF FNA(X)=A*X^2+B
DEF(type)	Defines variable types (types can be INT, SNG, DBL or STR)	10 DEFINT I-N,X 20 DEFSTR P
DEF USR	Defines starting address for machine language subroutine	10 DEF USR=8000
DIM	Defines (dimension) arrays	10 DIM X(100),Z$(20,100)
END	End of program	99 END
ERASE	Erases (eliminates) individual arrays	10 ERASE Z$
ERROR	Simulates the occurrence of an error	10 X=13 20 ERROR X
FIELD	Defines field length (random files)	10 FIELD 1,20 AS CUST$
FOR and NEXT	Define the start and end of a FOR-TO loop	10 FOR COUNT=1 TO 100 60 NEXT COUNT
GET(text mode)	Reads a record from a random file to a memory buffer	10 OPEN "R",#1,"DATA" 20 FIELD 1,20 AS CUST$ 30 GET 1

Statement	*Purpose*	*Example*
GET(graphics mode)	Stores portion of graphics screen display in an array	10 GET (10,10)-(80,50),SHAPE
GOSUB	Transfers control to a subroutine	10 GOSUB 200
GOTO	Transfers control to a remote statement	10 GOTO 200
IF-THEN	Conditional execution	10 IF X>0 THEN 200
IF-THEN-ELSE	Conditional execution	10 IF X>0 THEN 200 ELSE X=0
INPUT	Enters data from the keyboard	10 INPUT A,B,C,T$ 20 INPUT "X=",X
INPUT #	Enters data from a sequential file or device	10 OPEN "I",#1,"NAMES" 20 INPUT #1,N$
KEY OFF	Turns off function key display	10 KEY OFF
KEY ON	Turns on function key display	10 KEY ON
KILL	Deletes an entire file	10 KILL "SAMPLE"
LET	Assignment statement (optional)	10 LET X=A+B+C
LINE	Generates lines and rectangles (graphics mode)	10 LINE (0,0)-(319,199) 20 LINE (10,5)-(50,80),2,BF
LINE INPUT	Reads an entire line from the keyboard, as a string	10 LINE INPUT T$ 20 LINE INPUT;"Ans:";A$
LINE INPUT #	Reads an entire line from a sequential file as a string	10 OPEN "I",#1,"NAMES" 20 LINE INPUT #1,N$
LOCATE	Specifies the current cursor position (row and column)	10 LOCATE 12,40
LPRINT	Prints data on the printer	10 LPRINT A,B,C,T$ 20 LPRINT "X=";X
LPRINT USING	Prints formatted data on the printer	10 LPRINT USING "#.##";X
LSET	Places data into a random file buffer, left-justified	10 OPEN "R",#1,"DATA" 20 FIELD 1,20 AS CUST$ 30 LSET CUST$=NAME$
NAME	Renames a file	10 NAME "SAMPLE" AS "DATA"
ON ERROR GOTO	Transfers control if an error occurs	10 ON ERROR GOTO 500
ON-GOSUB	Transfers control to one of several subroutines	10 ON FLAG GOSUB 100,200,300
ON-GOTO	Transfers control to one of several destinations	10 ON K GOTO 80,120,160
ON KEY() GOSUB	Associates a function key with a subroutine	10 ON KEY(3) GOSUB 200 (refers to function key F3)
ON PEN GOSUB	Associates light pen activation with a subroutine	10 ON PEN GOSUB 300
ON STRIG() GOSUB	Associates a joystick button with a subroutine	10 ON STRIG(0) GOSUB 400
OPEN	Opens a file for input/output operations	10 OPEN "I",#1,"NAMES"

Statement	*Purpose*	*Example*
OUT	Sends a byte to an output port	10 OUT 127,3 (127 is the port no.)
PAINT	Fills an enclosed graphics shape with color	10 CIRCLE (160,100),10,2 20 PAINT (160,100),2,2
PEN OFF	Turns off light pen read function	10 PEN OFF
PEN ON	Turns on light pen read function	10 PEN ON
POKE	Places a value in a specified memory location	10 POKE(32155,65) (memory location is 32155, value is 65)
PRESET	Erases a point (graphics mode)	10 PRESET (25,40)
PRINT	Displays data on the screen	10 PRINT A,B,C,T$ 20 PRINT "X=";X
PRINT #	Writes data to a sequential file	10 PRINT #1,A;B;C;T$
PRINT USING	Displays formatted data on the screen	10 PRINT USING "#.##";X
PRINT # USING	Writes formatted data to a sequential file	10 PRINT #1, USING "#.##";X
PSET	Generates a point (graphics mode)	10 PSET(25,40),1
PUT(text mode)	Writes a record from a memory buffer to a random file	10 PUT#1,22 (22 is the record no.)
PUT(graphics mode)	Displays graphics image stored in an array	10 PUT (120,80),SHAPE
RANDOMIZE	Initializes the random number generator	10 RANDOMIZE
READ	Assigns values in DATA statement to listed variables	10 READ A,B,C,T$ 20 DATA 2.3,–0.1,6,RED
REM	Places remarks in the program	10 REM *** PROGRAM 1 ***
RESTORE	Initializes the pointer in a DATA statement	10 RESTORE 20 20 DATA 2.3,–0.1,6,RED
RESUME	Continues program execution after error correction routine	10 RESUME 100
RETURN	Used at end of subroutine; returns control to statement following GOSUB	10 GOSUB 80 80 REM BEGIN SUBROUTINE 100 RETURN
RSET	Places data into a random file buffer, right-justified	10 OPEN "R",#1,"DATA" 20 FIELD 1,20 AS CUST$ 30 RSET CUST$=NAME$
SCREEN	Specifies current mode (text or graphics)	10 SCREEN 0
SOUND	Generates a sound with a fixed frequency and duration	10 SOUND 800,100 (800=frequency, 100=duration)

Statement	*Purpose*	*Example*
STOP	Terminates program execution	10 STOP
STRIG OFF	Deactivates joystick buttons	10 STRIG OFF
STRIG ON	Activates joystick buttons	10 STRIG ON
SWAP	Exchanges the values of two different variables	10 SWAP X,Y
TIME$	Sets the current time	10 TIME$="13:07:42"
WAIT	Suspends program execution until a specified bit pattern is detected in an input port	10 WAIT 16,6 (input port=16)
WHILE and WEND	Define the start and end of a conditional loop	10 COUNT=1 20 WHILE COUNT<10 30 PRINT "COUNT=";COUNT 40 COUNT=COUNT+1 50 WEND
WIDTH	Specifies the number of characters per line	10 WIDTH 80
WRITE	Displays data on the screen (similar to PRINT)	10 WRITE A,B,C,T$
WRITE #	Writes data to a sequential file (similar to PRINT #)	10 WRITE #1,A,B,C,T$

Library Functions

Function	*Purpose*	*Example*
ABS	Returns absolute value	10 Y=ABS(X)
ASC	Returns ASCII code	10 Y=ASC(X$)
ATN	Returns arctangent	10 Y=ATN(X)
CDBL	Converts to double precision	10 Y#=CDBL(X)
CHR$	Returns character represented by given ASCII code	10 Y$=CHR$(X)
CINT	Converts to an integer	10 Y%=CINT(X)
COS	Returns the trigonometric cosine function	10 Y=COS(X)
CSNG	Converts to single precision	10 Y=CSNG(X#)
CSRLIN	Returns the vertical cursor position (line number)	10 Y=CSRLIN
CVD	Converts string to double precision value	10 Y#=CVD(X$)
CVI	Converts string to integer value	10 Y%=CVI(X$)
CVS	Converts string to real value	10 Y=CVS(X$)
DATE$	Returns the date	10 Y$=DATE$
EOF	Indicates an end-of-file	10 IF EOF(1) THEN 100
ERL	Returns the line number where an error occurred	(see next example)

Function	*Purpose*	*Example*
ERR	Returns an error code	10 PRINT ERR, ERL
EXP	Returns the exponential function	10 Y=EXP(X)
FIX	Converts to an integer (truncate)	10 Y%=FIX(X)
FRE	Returns the number of unused bytes of memory	10 Y=FRE(0)
HEX$	Converts from decimal to hexadecimal	10 Y$=HEX$(X)
INKEY$	Returns a character from the keyboard	10 Y$=INKEY$
INP	Returns a byte from an input port	10 Y=INP(127)
INPUT$	Returns a multicharacter string from the keyboard	10 Y$=INPUT$(3)
INSTR	Returns the position where one string (X$) is found within another string (T$)	10 Y=INSTR(T$,X$)
INT	Returns the largest integer that does not exceed the specified value	10 Y%=INT(X)
LEFT$	Returns the leftmost *n* characters of a string	10 Y$=LEFT$(X$, 3)
LEN	Returns the number of characters in a string	10 Y=LEN(X$)
LOC	Returns the current record number	10 Y=LOC(1)
LOF	Returns the file length, in bytes	10 Y=LOF(1)
LOG	Returns the natural logarithm	10 Y=LOG(X)
LPOS	Returns the column number of the current print-head position (for a printer)	10 IF LPOS>40 THEN PRINT "*"
MID$	Returns an *n*-character string, starting at location *m*	10 Y$=MID$(X$,5,3) (*m*=5, *n*=3)
MKD$	Converts double-precision value to a string	10 Y$=MKD$(X#)
MKI$	Converts integer value to a string	10 Y$=MKI$(X%)
MKS$	Converts real value to a string	10 Y$=MKS$(X)
OCT$	Converts from decimal to octal	10 Y$=OCT$(X)
PEN	Returns information associated with a light pen	10 X=PEN(1):Y=PEN(2)
PEEK	Returns the contents of a specified memory location	10 Y=PEEK(32155)
POINT	Returns the color of a point on the screen	10 Y=POINT(3,12)
POS	Returns the column number of the current cursor position	10 Y=POS(0)

Function	*Purpose*	*Example*
RIGHT$	Returns the rightmost *n* characters of a string	10 Y$=RIGHT$(X$, 3)
RND	Returns a random number between 0 and 1	10 Y=RND
SCREEN	Returns ASCII code for the character at the designated location	10 Y=SCREEN(5,12)
SGN	Returns an integer that indicates the sign of a value	10 Y=SGN(X)
SIN	Returns the trigonometric sine function	10 Y=SIN(X)
SPACE$	Returns a sequence of blank spaces	10 PRINT X;SPACE$(5);Y
SPC	Generates blank spaces in a PRINT statement	10 PRINT X;SPC(5);Y
SQR	Returns the square root of a value	10 Y=SQR(X)
STICK	Returns joystick coordinates	10 X=STICK(0):Y=STICK(1)
STRIG	Returns information associated with joystick buttons	10 Y=STRIG(0)
STR$	Converts a numerical value to a string	10 Y$=STR$(1000)
STRING$	Returns an *n*-character string of repeated characters	10 Y$=STRING$(8,42) (*n*=8, ASCII char=42)
TAB	Tabs to a specified position in a print statement	10 PRINT X;TAB(18);Y
TAN	Returns the trigonometric tangent	10 Y=TAN(X)
TIME$	Returns the current time	10 Y$=TIME$
USR	Accesses a machine-language subroutine	10 Y=USR(X)
VAL	Converts a string to a numerical value	10 Y=VAL(X$)
VARPTR	Returns the memory address of a variable	10 Y=VARPTR(X)

System Commands

Command	*Purpose*	*Example*
AUTO	Automatic line numbering	AUTO 100,10
CLEAR	Clears values assigned to numeric and string variables	CLEAR
CONT	Resumes program execution after a break	CONT
DELETE	Deletes program lines	DELETE 180–230
EDIT	Accesses a line for editing	EDIT 100
FILES	Displays names of all files	FILES
KILL	Deletes an entire file	KILL "SAMPLE"
LIST	Lists the program, or parts of the program, on the screen	LIST LIST 100–160

Command	*Purpose*	*Example*
LLIST	Lists the program, or parts of the program, on a printer	LLIST LLIST 100–160
LOAD	Loads a program into memory	LOAD "SAMPLE"
MERGE	Merges a program file into the program now in memory	MERGE "TRIAL"
NAME	Renames a file	NAME "SAMPLE" AS "NEWPROG"
NEW	Deletes the program currently in memory	NEW
RENUM	Renumbers program lines automatically	RENUM
RESET	Closes all files and clears the memory buffer	RESET
RUN	Initiates program execution	RUN
SAVE	Saves the program currently in memory	SAVE "SAMPLE"
SYSTEM	Exits from BASIC to the operating system	SYSTEM
TRACE ON (TRON)	Activates tracing of program statements during program execution	TRACE ON TRON
TRACE OFF (TROFF)	Discontinues tracing of program statements during program execution	TRACE OFF TROFF

Operators (listed hierarchically)

Operation	*Operator*
1. Exponentiation	↑ or ^
2. Negation	−
3. Multiplication and division	* /
4. Integer division	\
5. Integer remainder	MOD
6. Addition and subtraction	+ −
7. Relationals	= <> <= < >= >

Operation	*Operator*
8. Logical NOT	NOT
9. Logical AND	AND
10. Logical OR	OR
11. Logical XOR (exclusive OR)	XOR
12. Logical EQV (equivalence)	EQV
13. Logical IMP (implication)	IMP

Other Punctuation

Colon (:)	Used to separate statements on the same line. *Example*: 10 CLS : KEY OFF
Apostrophe (')	Used to designate comments on a statement line. *Example*: 10 CLS 'clear the screen

NOTES:

1. Many specific implementations include additional commands.
2. Some statements can also be used as system commands (e.g., CLS).
3. Some statements or functions may have different interpretations or multiple interpretations (e.g., GET, PUT).

ملحق (هـ)

The ASCII Character Set

ASCII Value	Character	ASCII Value	Character	ASCII Value	Character	ASCII Value	Character
000	NUL	032	blank	064	@	096	‘
001	SOH	033	!	065	A	097	a
002	STX	034	”	066	B	098	b
003	ETX	035	#	067	C	099	c
004	EOT	036	$	068	D	100	d
005	ENQ	037	%	069	E	101	e
006	ACK	038	&	070	F	102	f
007	BEL	039	’	071	G	103	g
008	BS	040	(	072	H	104	h
009	HT	041	)	073	I	105	i
010	LF	042	*	074	J	106	j
011	VT	043	+	075	K	107	k
012	FF	044	,	076	L	108	l
013	CR	045	-	077	M	109	m
014	SO	046	.	078	N	110	n
015	SI	047	/	079	O	111	o
016	DLE	048	0	080	P	112	p
017	DC1	049	1	081	Q	113	q
018	DC2	050	2	082	R	114	r
019	DC3	051	3	083	S	115	s
020	DC4	052	4	084	T	116	t
021	NAK	053	5	085	U	117	u
022	SYN	054	6	086	V	118	v
023	ETB	055	7	087	W	119	w
024	CAN	056	8	088	X	120	x
025	EM	057	9	089	Y	121	y
026	SUB	058	:	090	Z	122	z
027	ESC	059	;	091	[	123	{
028	FS	060	<	092	\	124	|
029	GS	061	=	093	]	125	}
030	RS	062	>	094	↑	126	~
031	US	063	?	095	—	127	DEL

Note: The first 32 characters and the last character are control characters; they cannot be printed.

رقم الإيداع ٧٠٩٦/٨٩

مطابع المكتب المصرى الحديث
MODERN EGYPTIAN PRESS
الكيلو ١٠ الطريق الزراعى مصر/اسكندرية ص ب ٣٥ شبرا الخيمة ت ٩٥٧٩٥٩

Summary of Standard BASIC Statements (Appendix A)

Statement	*Example*	*Reference*
CHANGE	10 CHANGE N$ TO N	Section 6.4
DATA	10 DATA 12,SEVENTEEN,−5	Section 5.5
DEF	10 DEF FNR(A,B,C)=SQR(A↑2+B↑2+C↑2)	Section 6.1
DIM	10 DIM A(10,20),X(20),F$(60)	Section 5.4
END	10 END	Section 2.11
FNEND	10 FNEND	Section 6.3
FOR-TO	10 FOR J=1 TO 99 STEP 2	Section 4.5
GO TO	10 GO TO 50	Section 2.14
GOSUB	10 GOSUB 300	Section 6.9
IF-THEN	10 IF I>=100 THEN 80	Section 4.2
INPUT	10 INPUT A,B,C,M$,N$	Section 2.9
LET	10 LET A=3.141593*R↑2	Section 2.8
NEXT	10 NEXT I	Section 4.6
ON-GO TO	10 ON K GO TO 15,40,25,40,60	Section 4.3
PRINT	10 PRINT "X=";X,"Y=";Y	Section 2.10
RANDOMIZE	10 RANDOMIZE	Section 6.7
READ	10 READ K,N$,Z(1)	Section 5.5
REM	10 REM AREA OF A CIRCLE	Section 2.13
RESTORE	10 RESTORE	Section 5.6
RETURN	10 RETURN	Section 6.8
STOP	10 STOP	Section 4.4

Statement	*Example*	*Reference*
MAT =	10 MAT C=A	Section 7.1
MAT +	10 MAT C=A+B	Section 7.1
MAT −	10 MAT C=A−B	Section 7.1
MAT (K)*	10 MAT C=(10)*A	Section 7.1
MAT *	10 MAT C=A*B	Section 7.1
MAT CON	10 MAT B=CON	Section 7.3
MAT IDN	10 MAT C=IDN	Section 7.3
MAT INPUT	10 MAT INPUT A	Section 7.2
MAT INV	10 MAT B=INV(A)	Section 7.3
MAT PRINT	10 MAT PRINT A	Section 7.2
MAT READ	10 MAT READ A	Section 7.2
MAT TRN	10 MAT B=TRN(A)	Section 7.3
MAT ZER	10 MAT A=ZER	Section 7.3
FILE	10 FILE :1,F$	Section 8.3
FILES	10 FILES SCORES	Section 8.1
IF END-THEN	10 IF END #1, THEN 130	Section 8.1
INPUT	10 INPUT #1,N,T$,Y$	Section 8.1
PRINT	10 PRINT #2,N;N$	Section 8.1
QUOTE	10 QUOTE #2	Section 8.1
READ	10 READ :1,L	Section 8.2
SCRATCH	10 SCRATCH #2	Section 8.1
SET (RESET)	10 SET :1,L	Section 8.2
WRITE	10 WRITE :1,N	Section 8.2

Arithmetic Operators: + − * / ↑

Relational Operators: = <> <= < >= >

Summary of Standard BASIC Library Functions (Appendix B)

Function	*Example*	*Reference*
ABS	10 LET Y=ABS(X)	Section 5.1
ATN	10 LET Y=ATN(X)	Section 5.1
ASC	10 LET N=ASC(T)	Section 6.5
CHR$	10 LET N$=CHR$(N)	Section 6.5
COS	10 LET Y=COS(X)	Section 5.1
COT	10 LET Y=COT(X)	Section 5.1
DET	10 LET X=DET	Section 7.3
EXP	10 LET Y=EXP(X)	Section 5.1
INT	10 LET Y=INT(X)	Section 5.1
LOC	10 LET N=LOC(1)	Section 8.2

Function	*Example*	*Reference*
LOF	10 LET N1=LOF(3)	Section 8.[illegible]
LOG	10 LET Y=LOG(X)	Section [illegible]
NUM	10 LET N(0)=NUM	Section [illegible]
RND	10 LET X=RND	Section [illegible]
SGN	10 LET Y=SGN(X)	Section [illegible]
SIN	10 LET Y=SIN(X)	Section 5[illegible]
SQR	10 LET Y=SQR(X)	Section 5.[illegible]
TAB	10 PRINT TAB(N);X	Section 5.1
TAN	10 LET Y=TAN(X)	Section 5.1

General Organization of the Alexandria Library

Summary of Standard BASIC Systems Commands (Appendix C)

Command	*Purpose*
BYE	Terminates timesharing session.
CATALOG	Lists names of all files being saved.
GOODBYE	Same as BYE.
LIST	Produces a listing of the current file.
NEW	Specifies that a new file will be created.
OLD	Accesses an existing file.
RENAME	Allows the name of the current file to be changed.

Command	*Purpose*
REPLACE	Causes the current file to be saved (stored) in place of the file previously stored with the same name. (The old file will be deleted.)
RUN	Causes the current program to be compiled and executed.
SAVE	Causes the current file to be saved (stored).
SCRATCH	Removes the current file from the computer's memory.
SYSTEM	Transfers control from BASIC to the system monitor.
UNSAVE	Cancels permanent storage of a file.

Microsoft BASIC is summarized in Appendix D, pp. 437-443